Responsible and Efficient Use of Web-Scale Data for Training Foundation Models

Pratyush Maini

CMU-ML-25-117
Machine Learning Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

September 2025

Thesis Committee:

Zico Kolter (Co-Chair) and Zachary Lipton (Co-Chair),
Aditi Raghunathan (CMU)
Nicolas Papernot (University of Toronto),
Ludwig Schmidt (Stanford University)

Submitted in partial fulfillment of the requirements for the Degree of Doctor of Philosophy

Copyright © 2025 Pratyush Maini

This research was sponsored by DARPA GARD Program and OpenAI Cybersecurity Award

Contents

Abstract 1

I Introduction 2

1 Introduction 3

II Data Attribution and Membership 6

2 LLM Dataset Inference 8
2.1 Introduction . . . 8
2.2 Problem Setup . . . 11
2.2.1 Metrics for LLM Membership Inference . . . 12
2.3 Failure of Membership Inference . . . 14
2.4 LLM Dataset Inference . . . 17
2.4.1 Procedure for the LLM Dataset Inference . . . 17
2.4.2 Assumptions for Dataset Inference . . . 19
2.4.3 Experimental Details . . . 19
2.4.4 Analysis and Results with Dataset Inference . . . 19

3 STAMP 24
3.1 Introduction . . . 24
3.2 Related Works . . . 27
3.2.1 Membership Inference Attacks (MIAs) . . . 27
3.2.2 Test-set Contamination Detection . . . 27
3.2.3 LLM Memorization . . . 28
3.3 Preliminaries . . . 28
3.3.1 Dataset Membership . . . 28
3.3.2 Watermarks for LLMs . . . 28
3.4 STAMP: Spotting Training Artifacts through Watermarked Pairs . . . 29
3.4.1 Watermarking Datasets . . . 29
3.4.2 Detecting Dataset Membership . . . 30
3.5 Experiments & Results . . . 31

3.5.1 Releasing Watermarked Test Sets . . . 32
3.5.2 Pretraining with Intentional Contamination . . . 33
3.5.3 Utility of Test Sets . . . 35
3.5.4 Parameters Affecting the Power of the Test . . . 35
3.6 Real World Case Studies . . . 37
3.7 Discussion . . . 40
3.8 Conclusion & Future Directions . . . 41

III Mechanisms and Localization of Memorization 42

4 Can Neural Network Memorization Be Localized? 44
4.1 Introduction . . . 44
4.2 Related Work . . . 47
4.3 Problem Setup . . . 48
4.4 Gradient Accounting . . . 49
4.4.1 Layer-wise norm contribution . . . 49
4.4.2 Clean and Noisy example gradient similarity . . . 50
4.5 Functional Criticality of Layers . . . 52
4.5.1 Layer Rewinding . . . 52
4.5.2 Layer Retraining . . . 54
4.6 Memorization by Individual Neurons . . . 54
4.6.1 How many neurons does it take to predict an example? . . . 55
4.6.2 Example-Tied Dropout: Directing Memorization to Specific Neurons 57
4.7 Memorization of Atypical Examples . . . 60
4.8 Discussion . . . 61

5 Rethinking LLM Memorization via Adversarial Compression 62
5.1 Introduction . . . 63
5.2 Do We Really Need Another Notion of Memorization? . . . 65
5.3 Measuring Memorization with Compression . . . 67
5.3.1 Definition . . . 67
5.3.2 MiniPrompt and Operationalizing Compressible Memorization . . . 68
5.3.3 Validation of MiniPrompt . . . 69
5.3.4 Illusion of Compliance . . . 72
5.4 Memorization in The Wild . . . 73
5.4.1 Harry Potter . . . 73

5.4.2 TOFU . . . 74
5.5 Conclusion . . . 75
5.6 Greedy Coordinate Gradient . . . 76
5.7 Random Search . . . 76
5.8 More Details of In-context Unlearning . . . 77

6 MAGIC: Diffusion Model Memorization Auditing via Generative Image Compression 78
6.1 Introduction . . . 79
6.2 Related Work . . . 81
6.3 Preliminaries . . . 83
6.3.1 Diffusion Models . . . 84
6.3.2 Memorization Definitions . . . 86
6.4 Brittleness of Existing Memorization Definitions . . . 87
6.4.1 Training-Time Interventions . . . 89
6.4.2 Inference-Time Interventions . . . 89
6.4.3 Experiments . . . 90
6.4.4 Implications and Motivation for New Metrics . . . 91
6.5 MAGIC: Memorization Auditing by Generative Image Compression . . . 91
6.5.1 Method Overview . . . 91
6.5.2 Soft Embedding Optimization for Image Compression . . . 92
6.5.3 Operationalizing MAGIC . . . 93
6.5.4 Experiments and Results . . . 94
6.6 How much copyrighted content do frontier models memorize? . . . 95
6.7 Conclusion . . . 96

7 MemSinks: Memorization-Aware Learning via Controlled Sinks 99
7.1 Introduction . . . 99
7.2 Channeling Training Dynamics with MemSinks . . . 102
7.2.1 Validation of MemSinksin TinyStories . . . 103
7.2.2 Larger Scale Experiments with MemSinks . . . 104
7.2.3 Practicality of MemSinks . . . 106
7.2.4 Exploring the Mechanism of MemSinks . . . 107
7.3 The Pitfalls of Post-Hoc Localization . . . 108
7.3.1 Experimental Setting . . . 108
7.3.2 Empirical Observations . . . 109
7.3.3 Theoretical Analysis of Mechanistic Entanglement . . . 110

7.4 The Mirage of Forced Localization . . . 112
7.4.1 Forcing Localization By Masking Gradients . . . 112
7.4.2 Empirical Findings . . . 112
7.4.3 Co-Adaptation of Model Components . . . 113

IV Unlearning and Training Dynamics 115

8 TOFU: Task of Fictitious Unlearning 117
8.1 Introduction . . . 117
8.2 New Task: Fictitious Author Question Answering . . . 118
8.2.1 The TOFU Dataset . . . 118
8.2.2 Evaluation Metrics . . . 119
8.3 Baseline Unlearning Methods and Results . . . 123
8.3.1 Baseline Results . . . 123
8.4 False Promise of Unlearning . . . 126
8.5 Discussion . . . 127

9 Open Unlearning 129
9.1 Introduction . . . 129
9.2 Overview of LLM Unlearning . . . 131
9.3 OpenUnlearning . . . 133
9.3.1 Design of OpenUnlearning . . . 134
9.3.2 Design of modules . . . 134
9.4 Evaluating Unlearning Evaluations . . . 136
9.4.1 Faithfulness . . . 136
9.4.2 Robustness . . . 138
9.4.3 Aggregation of Metrics . . . 140
9.4.4 Results and Discussion . . . 140
9.5 Benchmarking Unlearning Methods . . . 141
9.6 Conclusion . . . 142

10 SSFT: Second-Split Forgetting Time 145
10.1 Introduction . . . 145
10.2 Theoretical Results . . . 147
10.2.1 First-split Learning . . . 149
10.2.2 Second-split Forgetting . . . 149

10.3 Method . . . 150
10.3.1 Baseline Methods . . . 151
10.3.2 Example Characterization . . . 151
10.4 Empirical Investigation of First- and Second-Split Training Dynamics . . . 152
10.4.1 Experimental Setup . . . 152
10.4.2 Learning-Forgetting Spectrum for various datasets . . . 153
10.4.3 Ablation Experiments . . . 154
10.4.4 Dataset Cleansing . . . 155
10.4.5 Evaluating Example Utility . . . 157
10.4.6 Characterizing Potential Failure Modes . . . 158
10.5 Conclusion . . . 159

11 T-MARS **160**
11.1 Introduction . . . 160
11.2 Method . . . 160
11.2.1 T-MARS: Text-Masking and Re-Scoring . . . 161
11.2.2 Contributed Baselines . . . 162
11.2.3 Existing Baselines . . . 162
11.3 Experiments . . . 163
11.3.1 Data Pools and Training Configuration . . . 163
11.3.2 Evaluation Datasets . . . 163
11.3.3 Results . . . 165
11.3.4 T-MARSeffectively filters targeted images . . . 166
11.4 Limitations and Conclusion . . . 167

12 FADU **168**
12.1 Introduction . . . 168
12.2 Scaling Laws for Data Filtering . . . 169
12.2.1 Defining Utility . . . 169
12.2.2 Utility under Repetition . . . 169
12.2.3 The case of heterogeneous web data . . . 170
12.2.4 Various other formulations . . . 171
12.2.5 Scaling the scaling curves . . . 172
12.3 Discussion . . . 173

V Synthetic Data and Safety 174

13 Rephrasing the Web: WRAP 176
13.1 Introduction . . . 176
13.2 T-MARS: Web Rephrase Augmented Pretraining . . . 177
13.2.1 Rephrasing the Web . . . 177
13.2.2 Implementation Details . . . 178
13.3 Perplexity Evaluation . . . 179
13.4 Zero-shot Tasks . . . 180
13.4.1 Datasets . . . 180
13.4.2 Results . . . 181
13.5 Analysis and Ablations . . . 182
13.6 Rephrase Examples from C4 Corpus . . . 184
13.7 Properties of Synthetic Corpus . . . 185
13.7.1 Experimental Setup . . . 186

14 BeyondWeb 187
14.1 Introduction . . . 187
14.2 A Tale of Two Approaches for Synthetic Pretraining Data . . . 190
14.2.1 The Generator-Driven Paradigm: Creating Knowledge from Models *de Novo* . . . 190
14.2.2 The Source Rephrasing Paradigm: Enhancing Existing Knowledge . 190
14.2.3 Synthetic Data Beyond Pretraining . . . 191
14.3 Introducing BeyondWeb . . . 192
14.4 Systematically Evaluating Synthetic Data . . . 195
14.4.1 Experiment Setup . . . 195
14.4.2 Research Question (RQ) 1: Are Generator-Driven Approaches Approximated by Summarization? . . . 197
14.4.3 RQ2: Can Synthetic Data Break the Data Wall? . . . 199
14.4.4 RQ3: How Important is the Quality of the Seed Data for Rephrasing? 201
14.4.5 RQ4: Is Distributional Style Matching Important? . . . 203
14.4.6 RQ5: How Important is Diversity at Scale? . . . 205
14.4.7 RQ6: How Important is the Synthetic Rephraser Model Family? . . . 207
14.4.8 RQ7: Does Rephraser Size Matter? . . . 209
14.5 Future Directions . . . 210
14.6 Conclusion . . . 211
14.7 Contributions and Acknowledgements . . . 212

15 Understanding Hallucinations in Diffusion Models 213
15.0.1 Hallucination in Diffusion Models . . . 213
15.1 Definitions and Preliminaries . . . 215
15.2 Understanding Mode Interpolation and Hallucination . . . 216
15.2.1 Gaussian-1DSetup . . . 216
15.2.2 Gaussian-2DGrid . . . 217
15.2.3 What causes mode interpolation? . . . 218
15.2.4 Squares-Triangles . . . 219
15.2.5 Mode Interpolation in Real World datasets: Hands . . . 219
15.3 Diffusion Models know when they Hallucinate . . . 221
15.3.1 Variance in the trajectory of prediction . . . 221
15.3.2 Metric for detecting hallucination . . . 222
15.4 Implications on Recursive Model Training . . . 223
15.5 Discussion . . . 226

16 Safety Pretraining 227
16.1 Introduction . . . 227
16.2 Related Work . . . 229
16.3 Pre-training Data Interventions . . . 230
16.3.1 Safety Scoring . . . 230
16.3.2 Synthetic Recontextualization . . . 231
16.3.3 Refusing the Web . . . 232
16.3.4 Moral Education Data . . . 233
16.3.5 Data Safety Report Cards, A New Standard . . . 233
16.4 Evaluations & Results . . . 234
16.4.1 Performance on Standard Benchmarks . . . 234
16.4.2 Safety Evaluations . . . 235
16.4.3 Helpfulness and Overrefusal Evaluations . . . 237
16.5 Discussion . . . 238
16.6 Ablations . . . 238
16.6.1 Importance of Various Data-Centric Interventions . . . 238
16.6.2 <potentially_unsafe_content> Annotation During Pretraining vs. Finetuning . . . 240
16.7 Future Work and Conclusion . . . 241

A LLM Dataset Inference 243

B STAMP **251**

C TOFU: Task of Fictitious Unlearning **269**

D OpenUnlearning **288**

E SSFT **315**

F FADU **335**

G Diffusion Model Hallucinations **348**

H Rephrasing the Web (WRAP) **354**

I BeyondWeb **377**

Abstract

Foundation models—large neural networks trained on web-scale data—have transformed AI, but their use of massive internet corpora raises two critical challenges. First, responsible data use: models sometimes memorize and regurgitate sensitive or copyrighted content, posing privacy and legal risks. Ensuring proper attribution of training data and the ability to unlearn specific data post-hoc are thus paramount. Second, efficient data use: indiscriminately scaling up data and model size is computationally prohibitive, especially as high-quality web data becomes scarce. Training must be made more efficient through judicious data curation and augmentation.

This thesis addresses both challenges through novel contributions organized in two themes: (1) Understanding and Mitigating Memorization in Foundation Models, and (2) Enhancing Data Quality and Training Efficiency at Scale. In Theme (1), we develop techniques to attribute model outputs to training data, detect and localize memorized knowledge in model parameters, and devise frameworks for unlearning content to prevent privacy leaks. In Theme (2), we introduce methods to characterize and filter web-scale data for maximum utility, establish scaling laws that guide the curation of training sets under compute constraints, and leverage synthetic data generation to dramatically speed up model pre-training. By combining these efforts, the thesis proposes a path toward foundation models that learn from web-scale data both responsibly—safeguarding privacy and data rights, and efficiently—extracting maximum performance from each unit of data and compute.

Part I

Introduction

Chapter 1

Introduction

The past few years have seen the rise of **foundation models**, exemplified by systems like GPT-3 [Brown et al., 2020b] and CLIP, which are trained on unprecedented quantities of internet text and images. Scaling up model parameters (e.g. to 175 billion) and training on diverse web-scraped corpora has yielded remarkable cross-domain capabilities [Brown et al., 2020b]. These models can perform zero-shot question answering, translation, image recognition, and more, without task-specific training. **However, their very reliance on web-scale data brings new challenges.** Unlike curated datasets used in traditional supervised learning, web data is *unstructured, noisy, and rife with privacy and copyright sensitive content* [Schwarzschild et al., 2024b]. This raises urgent questions about how to responsibly source and utilize such data. On the one hand, more data can mean better performance; on the other, *indiscriminate use of raw web data can embed harmful memorization or waste computation on low-quality content*.

A key concern is **memorization of training data**. When exposed to billions of tokens, a model inevitably learns rare or unique sequences verbatim. Researchers have shown that large language models can unintentionally regurgitate personal information or copyrighted text from their training sets [Schwarzschild et al., 2024b] [Carlini et al., 2021c]. For example, GPT-2 was found to memorize over 600 snippets of its training data, including private details, which could be extracted with clever prompts [Carlini et al., 2021c]. Repeating data during training greatly amplifies memorization [Carlini et al., 2021c]. This poses privacy risks and challenges compliance with regulations (e.g. GDPR's "right to be forgotten"). Simply removing all sensitive data before training is impractical at web scale, as sensitive content is hard to reliably identify in advance. Thus, *techniques for detecting and managing memorization after training are needed*. Equally important is attributing a model's knowledge to its data sources: without proper attribution, content creators cannot be credited or compensated, and misuse of unlicensed data goes unchecked. High-profile cases (e.g. artists finding their work reproduced by image models) highlight the need for data provenance in model training.

Another challenge is **data quality and scaling efficiency**. Web-scraped corpora contain vast amounts of noise – spam, duplications, poorly formatted text, or images with trivial correlations. Training on such data is wasteful: it consumes enormous compute while

yielding marginal benefit or even impairing generalization. Recognizing this, recent efforts like LAION-5B applied aggressive filtering, discarding ~90% of scraped image-caption pairs to keep only those with high image–text alignment [Schuhmann et al., 2022]. But *how best to curate data for training remains an open question*, especially under different compute budgets. Using only the "highest-quality" data can under-utilize available compute (once the model has seen that data enough times, additional epochs yield diminishing returns [Goyal et al., 2024]), whereas including more data of lower quality might boost performance if training longer [Goyal et al., 2024]. This trade-off implies that data curation should be *compute-aware*. Furthermore, as model sizes grow, current scaling laws (e.g. the Chinchilla optimal balance between model and data size) demand ever larger training sets – yet truly high-quality data may soon run out [Maini et al., 2024d]. Generating or augmenting data in a way that *increases effective dataset size without proportional cost* is therefore crucial.

In summary, the twin goals of **responsibility** and **efficiency** in foundation model training are often in tension with the scale of data involved. This thesis proposal outlines research to tackle these goals via two themes:

- **Memorization in Foundation Models (Responsible Data Use):** How can we *detect, quantify, and mitigate memorization* of training data in large models? We investigate methods to attribute model outputs to training data sources, techniques to localize memorized knowledge inside models, and algorithms to *unlearn* or remove specific data influences post-training. By doing so, we aim to prevent privacy leaks and ensure models do not retain data they shouldn't, thereby enabling more accountable use of web-scale data.

- **Data Quality and Training Efficiency (Efficient Data Use):** How can we *maximize model performance per unit of data and compute*? We develop metrics to characterize the value of data (identifying mislabeled or uninformative examples), new data filtering algorithms to curate better training sets, scaling laws that inform optimal data usage under compute constraints, and approaches to augment training data with *synthetic but informative* examples. These techniques reduce the needed training compute and mitigate the environmental and financial costs of foundation model training.

We organize the thesis into six chapters, each centered on one or more of our contributions in these areas. **Chapter 1** covers dataset attribution and provenance, introducing our method to trace a model's training data and a large-scale audit of licensing in AI datasets. **Chapter 2** delves into the mechanisms of memorization, defining new metrics for memorization and pinpointing where networks store memorized knowledge. **Chapter 3** discusses strategies

for *unlearning* and memorization-aware training, including a benchmark for forgetting in language models and techniques to identify and remove memorized content during training. Shifting to data quality and efficiency, **Chapter 4** presents a filtering approach that improves data quality by removing unhelpful examples (e.g. vision data where captions spuriously match images). **Chapter 5** introduces scaling laws for data curation, guiding how to allocate a fixed compute budget across data of varying quality. **Chapter 6** explores the use of synthetic data to reduce required training data and speed up pre-training. Finally, we conclude with **Future Work**, outlining open directions for building *safe and efficient* training pipelines for the next generation of foundation models. The remainder of this proposal details each chapter's theme, contributions, and key results.

Part II

Data Attribution and Membership

Overview

The rapid adoption of large-scale generative models has intensified questions of data provenance, licensing, and accountability. Beyond isolated anecdotes, stakeholders increasingly need principled tools to determine whether and how specific datasets influenced a model. This Part develops two complementary perspectives: we first diagnose why instance-level membership inference can be misleading under distribution shift, and then advance dataset-level attribution that better reflects real legal and practical claims. We further introduce watermark-based dataset instrumentation that empowers creators and benchmark curators to attest inclusion with strong statistical evidence while preserving utility.

We aim to make responsible data use actionable by providing detection methods that are robust, sample-efficient, and resistant to common confounders. Together, these chapters establish a pragmatic foundation for audits and disclosure.

Guiding questions

- Q1: What are the limits of instance-level membership inference under realistic distribution shifts?
- Q2: When does dataset-level inference provide stronger, more reliable attribution signals?
- Q3: How can we instrument datasets (e.g., via rephrasing + watermarks) to enable auditable, statistically valid inclusion tests?
- Q4: How many queries and what signal combinations are needed to minimize false positives while retaining power?

Chapter 2

LLM Dataset Inference

Based on: Maini et al. [2024b]: Pratyush Maini, Hengrui Jia, Nicolas Papernot, Adam Dziedzic. *LLM Dataset Inference*. NeurIPS 2024

Abstract

The proliferation of large language models (LLMs) in the real world has come with a rise in copyright cases against companies for training their models on *unlicensed data* from the internet. Recent works have presented methods to identify if individual text sequences were members of the model's training data, known as membership inference attacks (MIAs). We demonstrate that the apparent success of these MIAs is confounded by selecting non-members (text sequences not used for training) belonging to a different distributions from the members (e.g., temporally shifted recent Wikipedia articles compared with ones used to train the model). This distribution shift makes membership inference appear successful. However, most MIA methods perform no better than random guessing when discriminating between members and non-members from the same distribution (e.g., in this case, the same period of time). Even when MIAs work, we find that different MIAs succeed at inferring membership of samples from different distributions. Instead, we propose a new *dataset inference* method to accurately identify the datasets used to train large language models. This paradigm sits realistically in the modern-day copyright landscape, where authors claim that an LLM is trained over multiple documents (such as a book) written by them, rather than one particular paragraph. While dataset inference shares many of the challenges of membership inference, we solve it by selectively combining the MIAs that provide positive signal for a given distribution, and aggregating them to perform a statistical test on a given dataset. Our approach successfully distinguishes the train and test sets of different subsets of the Pile with statistically significant p-values < 0.1, and no false positives.

2.1 Introduction

Training of large language models (LLMs) on large scrapes of the web [Google DeepMind, 2025, OpenAI, 2025] has recently raised significant privacy concerns [Rahman and Santacana, 2023, Wu et al., 2023]. The inclusion of personally identifiable information (PII)

and copyrighted material in the training corpora has led to legal challenges, notably the lawsuit between *The New York Times* and OpenAI [Grynbaum and Mac, 2023a], among others [Baker Law, 2023, Grynbaum and Mac, 2023b]. Such cases highlight the issue of using copyrighted content without attribution and/or license. Potentially, they undermine the rights of creators and disincentivize future artistic endeavors due to the lack of monetary compensation for works freely accessible online. This backdrop sets the stage for the technical challenge of identifying training data within machine learning models [Shokri et al., 2017b, Maini et al., 2021]. Despite legal ambiguities, the task holds critical importance for understanding LLMs' operations and ensuring data accountability.

Membership inference [Shokri et al., 2017b] is a long-studied privacy problem, intending to infer if a given data point was included in the training data of a model. However, identifying example membership is a challenging task even for models trained on small datasets [Carlini et al., 2022b, Duan et al., 2023], and Maini et al. [2021] presented an impossibility result suggesting that as the size of the training set increases, the success of membership inference degrades to random chance. *Is testing the membership of individual sentencesfor LLMs trained for a single epoch on trillions of tokens of text data feasible?* In our work, we first demonstrate that previous claims of successful membership inference for individual text sequences in LLMs [Mattern et al., 2023, Shi et al., 2024a] are overly optimistic (Section 2.3). Our evaluation of the MIA methods for LLMs reveals a crucial confounder: they detect (temporal) distribution shifts rather than the membership of data points (as also concurrently observed by [Duan et al., 2024b]). Specifically, we find that these MIAs infer whether an LLM was trained on a *concept* rather than an individual *sentence*. Even when the outputs of such MIAs (weakly) correlate with actual sentencemembership, we find that they remain very brittle across sentencesfrom different data distributions, and no single MIA succeeds across all. Based on our experiments, we conclude the discussion of MIAs with guidelines for future researchers to conduct robust experiments, highlighting the importance of using IID splits (between members and non-members), considering various data distributions, and evaluating false positives to mitigate confounding factors.

If membership inference attacks are so brittle, do content writers and private individuals have no recourse to claim that LLM providers unfairly trained on their data? As an alternative to membership inference, we advocate for a shift in focus towards dataset inference [Maini et al., 2021], which is a statistically grounded method to detect if a given *dataset* was in the training set of a model. We propose a new dataset inference method for LLMs that aims at detecting sets of text sequences by specific authors, thereby offering a more viable approach to dataset attribution than membership inference. Our method is presented in Figure 2.1. The motivation behind dataset inference stems from the

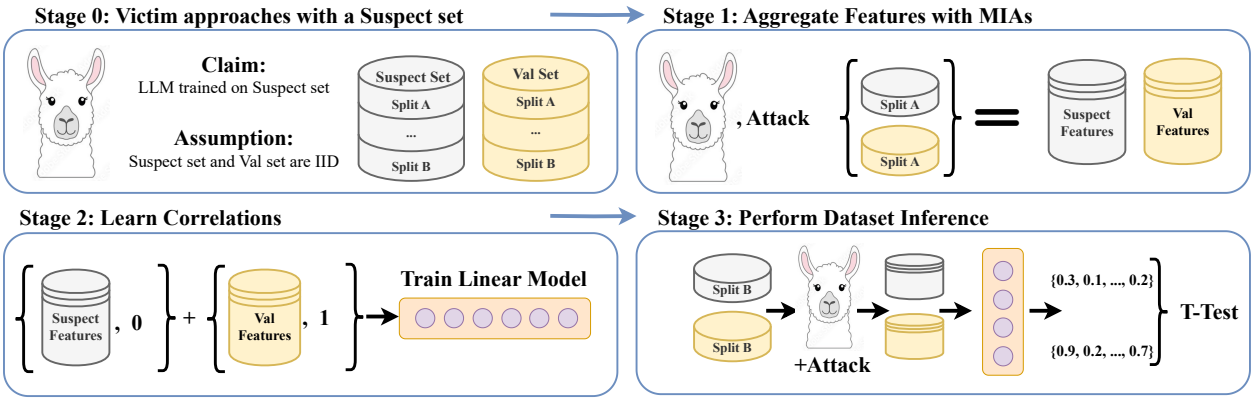


Figure 2.1: LLM Dataset Inference. ***Stage 0:*** *Victim approaches an LLM provider.* The victim's data consists of the suspect and validation (Val) sets. A victim claims that the suspect set of data points was potentially used to train the LLM. The validation set is the victim's unpublished data (e.g., drafts of articles, blog posts, or books) from the same distribution as the suspect set. Both sets are divided into non-overlapping splits (partitions) A and B. ***Stage 1:*** *Aggregate Features with MIAs.* The A splits from suspect and validation sets are passed through the LLM to obtain their features, which are scores generated from various MIAs for LLMs. ***Stage 2:*** *Learn Correlations (between features and their membership status).* We train a linear model using the extracted features and assigning label 0 (denoting potential members of the LLM) to the suspect and label 1 (representing non-members) to the validation features. ***Stage 3:*** *Perform Dataset Inference.* We use the B splits of the suspect and validation sets, pass them through the suspect LLM to obtain features, then do inference on the trained linear model to obtain the confidence scores, and apply a statistical T-Test on the obtained scores. For the suspect data points that are members, their confidence scores are significantly closer to 0 than for the non-members.

observation that in the rapidly evolving discourse on copyright, individual data points have much less agency than sets of data points attributed to a particular creator; and the fact that more often than not, cases of unfair use emerge in scenarios when multiple such sequences or their clusters naturally occur. For instance, consider the Harry Potter series written by J.K. Rowling. Dataset inference tests whether a 'dataset' or a collection of paragraphs from her books was used for training a language model, rather than testing the membership of individual sentencesalone. We also outline the specific framework required to operationalize dataset inference, including the necessary assumptions for the same.

We carry out our analysis of dataset inference using LLMs with known training and validation data. Specifically, we leverage the Pythia suite of models [Biderman et al., 2023a] trained on the Pile dataset [Gao et al., 2020] (Section 2.4). This controlled experimental setup allows us to precisely analyze the model behavior on members and non-members when they occur IID (without any temporal shift) as the training and validation splits of PILE are publicly accessible. Across all subsets, dataset inference achieves p-values less than 0.1 in distinguishing between training and validation splits. At the same time, our method shows no false positives, with our statistical test producing p-values larger than 0.5 in all cases when

comparing two subsets of validation data. To its practical merit, dataset inference requires only 1000 text sequences to detect whether a given suspect dataset was used to train an LLM.

2.2 Problem Setup

LLMs train on trillions of tokens, and the sizes of the training sets are only likely to increase [Meta AI, 2025, Databricks, 2025]. To increase training efficiency (in terms of time, financial costs, and environmental impact), improve performance, and decrease the risk of privacy leakage, many LLM practitioners deduplicate their pre-training data [Biderman et al., 2023a, Carlini et al., 2021b, Lee et al., 2022]. In our work, we ask this question: *How to detect if a given dataset was used to train an LLM?* and propose the idea of dataset inference for LLMs.

Access Levels. In the *black-box* setting, we assume an input-output access to an LLM along with access to model loss, hence we are not allowed to inspect individual weights or hidden states (*e.g.*, attention layer parameters) of the language model. This threat model is realistic in the case of LLM's users since many language models can be accessed through APIs that provide limited visibility into their inner workings. For instance, OpenAI [OpenAI, 2025] offers API access to GPT-3 and GPT-4, while Google [Google DeepMind, 2025] offers Gemini, without revealing the full architecture of the models or training methodology. The *gray-box* access, commonly assumed for MIAs, additionally assumes that we can obtain the perplexity or the loss values from an LLM, however, no additional information such as model weights or gradients. In the *white-box* access, we assume full access to the model, where we can inspect model weights.

Operationalizing Dataset Inference. Dataset inference for LLMs serves as a detection method for data used to train an LLM. We consider the following three actors during a dispute:

1. *Victim ($\mathcal{V}$).* We consider a victim creator whose private or copyrighted content was used to train an LLM without explicit consent. The actor is presumed to have only black-box access to the suspect model, which limits their ability to evaluate if their dataset was used in the LLM's training process.
2. *Suspect ($\mathcal{A}$).* The suspect (or potential adversary in this case) is an LLM provider who may have potentially trained their model on the victim's proprietary, or private data.
3. *Arbiter.* We assume the presence of an arbiter, *i.e.*, a third-trusted party, such as law enforcement, that executes the dataset inference procedure. The arbiter can obtain gray-

box access to the suspect LLM. For instance, in scenarios when API providers only give black-box access to users, legal arbiters may have access to model loss to perform MIAs.

Scenario. Consider a scenario where a book writer discovers that their publicly available but copyrighted manuscripts have been used without their consent to train an LLM. The writer (the *victim*, $\mathcal{V}$) gathers a small set of text sequences (say 100) from their manuscripts that they believe the model was trained on. The *suspect* $\mathcal{A}$ in this scenario is the LLM provider, who may have included the writer's published work in their training data without obtaining explicit permission. The provider is under suspicion of potentially infringing on the writer's manuscripts. An *arbiter*, such as a law enforcement agency, steps in to resolve the dispute. The arbiter obtains gray-box access to the suspect LLM, allowing them to execute our dataset inference procedure and resolve the dispute. By performing dataset inference (as depicted in Figure 2.1), the arbiter determines whether the writer's published manuscripts were used in the training of the LLM. This process highlights the practical application and significance of dataset inference in safeguarding the rights of artists.

Notation. We consider x to be an input sentence with N tokens $x = x_1, x_2, ..., x_N$ and $f$is a Language Model (LM) with parameters θ. We can compute the probability of an arbitrary sequence $p(x_1, ..., x_n)$, and obtain next-token x_{n+1} predictions. For simplicity, assume that the next token is sampled under greedy decoding, as the next token with the highest probability given the first n tokens.

2.2.1 Metrics for LLM Membership Inference

This section explores various metrics used to assess Membership Inference Attacks (MIAs) against LLMs. We study MIAs under gray-box access (which assumes access to the model loss, but not to parameters or gradients). The adversary aims to learn an *attack* function $A_p : \mathcal{X} \to \{0, 1\}$ that takes an input x from distribution $\mathcal{X}$ and determines whether x was in the training set $\mathcal{D}_{\text{train}}$of the LM f_θ or not. Let us now describe the MIAs we use in our work.

Thresholding Based. These MIAs leverage loss [Yeom et al., 2018a] or perplexity [Carlini et al., 2021b] as scores and then threshold them to classify samples as members or non-members. Specifically, the decision rule for membership is: $A_p(x) = \mathbb{1}[\mathcal{L}(p, x) < \gamma]$, where γ is a selected pre-defined threshold. However, MIAs based solely on perplexity suffer from many false positives, where simple and predictable sequences that never occur in the training set can be labeled as *members*.

Min-k% Prob. As a remedy to the problem of predictability, Shi et al. [2024a] proposed the Min-k% Probmetric which evaluates the likelihood of the $K\%$ of tokens in x that have the lowest probability, conditioned on the preceding tokens. Hence, this MIA ignores highly predictable tokens in the suspect sequence. The membership prediction is made by thresholding the average negative log-likelihood of these tokens. The input sentence x is marked as included in pretraining data simply by thresholding the Min-k% Probresult: $A_p(x) = \mathbb{1}[\text{Min-k\% Prob}(x) < \gamma]$.

Perturbation Based. The central hypothesis behind Perturbation-based MIAs is that a sample that an LLM saw during training should have a lower perplexity on its original version (x), as opposed to a perturbed version of the same ($\tilde{x}$). Formally, the membership attack is defined as $A_p(x) = \mathbb{1}[\mathcal{P}_{p(x)}/\mathcal{P}_{p(\tilde{x})} < \gamma]$, for a threshold γ. In our work, we investigate various forms of perturbations such as (1) white-space perturbation, (2) synonym substitution [Mattern et al., 2023], (3) character-level typos, (4) random deletion, and (5) changing character case.

DetectGPT. This is a special case of perturbation-based MIAs, originally designed to detect machine-generated text [Mitchell et al., 2023]. The key difference is that perturbations to the input are made using an external language model that infills randomly masked-out spans of the original input. It then compares the log-probability of x with expected value of the same from multiple infilled neighbors $\tilde{x}_i$.

Reference Model Based. These methods compare the perplexity ratio between a suspect model and a reference model on a given string. The suspect model may have seen the string during training, while the reference model has not. The corresponding MIA is: $A_p(x) = \mathbb{1}[\mathcal{L}(p, x) < \mathcal{L}(p', x)]$, where p' is the reference model. In our work, we use the SILO [Min et al., 2023], Tinystories-33M [Eldan and Li, 2023a], Tinystories-1M [Eldan and Li, 2023a], and Phi-1.5 [Li et al., 2023d] models as reference models. Notably, these models were not trained on general web data. In particular, the Phi-1.5 and Tinystories models were trained on synthetic data generated by GPT models, and the SILO model was trained on data that is freely licensed for training.

zlib Ratio. Another simple MI baseline uses the *zlib library* [Gailly and Adler, 2004], where a potential member has a low ratio of the model's perplexity to the entropy of the text, which is computed as the number of bits for the sequence when compressed with the zlib library: $A_p(x) = \mathbb{1}[\mathcal{P}_p(x)/zlib(x) < \gamma]$ [Carlini et al., 2021b]. The idea is that a model

trained on a dataset will have low perplexity for its members because it was optimized for them, unlike the zLib algorithm, which was not tailored to the training data.

2.3 Failure of Membership Inference

We demonstrate that the challenge of successfully performing membership inference for large language models (LLMs) remains unresolved. This problem is inherently difficult because LLMs are typically trained for a *single epoch* on trillions of tokens of web data. In their work, Maini et al. [2021] demonstrated a near impossibility result (Theorem 2), suggesting that as the size of the training set increases, the success rate of any MIA approaches 0.5 (as good as a coin flip). While this was shown in a simplified theoretical model, we assess how this holds up for contemporary LLMs with billions of parameters. As a demonstrative example, we consider the most recent (and supposedly best performing) work that proposed the Min-k% Prob [Shi et al., 2024a] membership inference attack, alongside a dedicated dataset to facilitate future evaluations. In their work, they show that this method performs notably better than other MIAs such as perplexity thresholding and DetectGPT that they benchmark their work against.

Temporal Shift and the Need for IID Analysis. The evaluation dataset used to showcase the success of Min-k% Probwas the WikiMIA dataset, a dataset constructed using spans of Wikipedia articles written before (train set) and after the cut-off year 2023 (validation set). This was chosen considering the training of the Pythia models [Biderman et al., 2023a], which was based on scrapes of Wikipedia before 2023. Note that such an evaluation setup naturally has the potential confounder of a temporal shift in the concepts in data before and after 2023. Any article written after 2023 was naturally a non-member of the Pythia models, and those written before 2023 were considered members. However, with changing times, we also encounter temporal shifts in writing styles and concepts in the Wikipedia dataset. This raises concerns if membership tests using WikiMIA actually assess membership of a particular data point, or of that concept/style. A similar question was concurrently asked by Duan et al. [2024b], who independently showed that MIAs are only successful because of the temporal shift in such datasets.

To critically assess the robustness of the Min-k% Probmethod, we conducted an exploration using the Pythia models and their (original) train and validation splits that come from the PILE [Gao et al., 2020] dataset, as provided by the authors during pre-training. This facilitates a confounder-free evaluation of the capabilities of membership inference attacks. In particular, the PILE dataset has more than 20 different domain-specific subsets with

their own training and validation splits, such as Arxiv, Wikipedia, OpenWebtext to name a few. Some of the key observations from our experiments on the PILE were (Figure 2.2a):

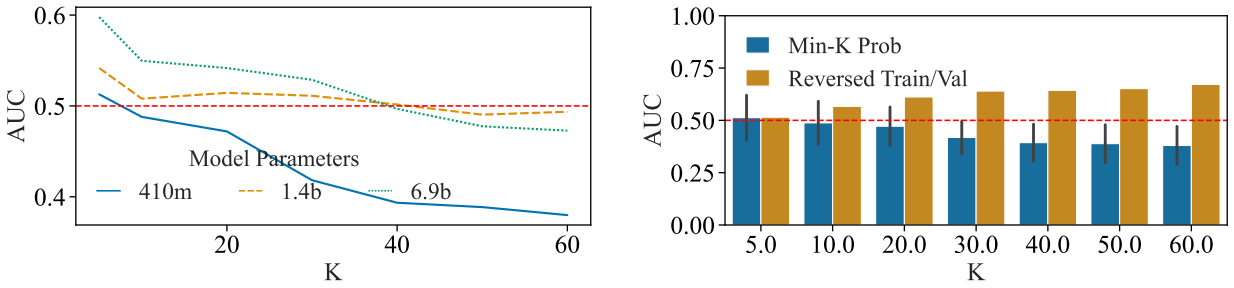
1. Contrary to performance on the WikiMIA dataset where Min-k% Probmetric achieved an AUC close to 0.7, the method got an AUC close to 0.5 when tested on IID train and validation splits of Wikipedia from the PILE dataset, hinting at a performance akin to random guessing.

2. We found that the method shows very high variance in AUC between different random subsets of the training and validation sets of the PILE dataset, oscillating between 0.4 and 0.7.

3. Results on Arxiv and OpenWebText2 subsets of the PILE show AUC values near 0.4, suggesting that Min-k% Probsuffers from false positives, labeling validation set examples as members.

False Positive Assessment by Reversing Train and Val Sets. *Do membership inference attacks actually test membership?* To answer this question, we do the following modification to the WikiMIA setup: For every sentence in the pre-2023 subset of Wikipedia, we replace it with a sentence from the validation set for Wikipedia as given in the PILE dataset. We keep the post-2023 Wikipedia subset as it is. On the one hand, since Pythia models were trained before 2023, it is clear that they never trained on data on Wikipedia from pages written after 2023. On the other hand, we also know that the validation set of the PILE dataset was not trained on and was also deduplicated from the train set. We now perform the same membership inference test on these two data splits Wikipedia Val (as the now designated 'suspect' set) versus Wikipedia post-2023 (as the supposed unseen set). Remarkably, the method demonstrates an extremely high AUC of 0.7 in labeling examples from the suspect (validation set) as members of the training set (Figure 2.2b). This confirms that these membership inference attacks (such as Min-k% Prob) only distinguish between concepts across different temporal phases rather than verifying specific data membership, which they were originally designed for.

No single MIA works across distributions. Now, we further expand our experimentation across multiple different membership inference attacks outlined in Section 2.2.1, across 20 different subsets of the PILE dataset. The goal is to analyze if there is any MIA that consistently performs well across all such distributions. In Figure 2.3, we show a heatmap of the performance of various (selected) MIA methods across different distributions of the PILE (refer to Appendix A for full results). While some MIAs perform well and achieve



(a) Performance for different model sizes.

(b) False Positives when reversing train/val sets.

Figure 2.2: Comparative analysis of the MIN-K% PROB [Shi et al., 2024a]. We measure the performance (a) across different model sizes and (b) the observed reversal effect. The method performs close to a random guess on non-members from the Pile validation sets.

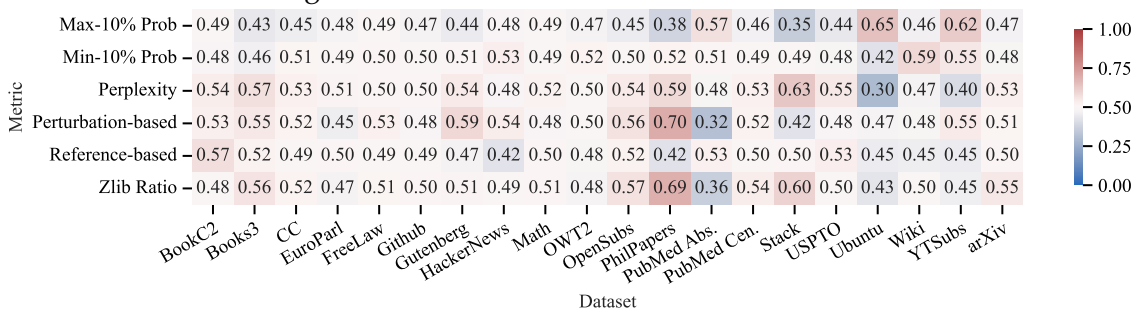


Figure 2.3: Performance of various MIAs on different subsets of the Pile dataset. We report 6 different MIAs based on the best performing ones across various categories like reference based, and perturbation based methods (Sec. 2.2.1). An effective MIA must have an AUC much greater than 0.5. Few methods meet this criterion for specific datasets, but the success is not consistent across datasets.

high AUC (such as synonym substitution on PhilPapers), the same methods have an AUC of less than 0.5 on the next dataset of Pubmed Abstracts. These results consolidate the finding that no single MIA for LLMs works across all datasets, and we need to potentially find methods that adapt the choice of metric to the distribution. In Section 2.4, we will leverage a (selective) combination of different MIAs to improve over the performance of any single MIA in order to perform successful LLM Dataset Inference.

Guidelines for Future Research. Based on our observations in this section, we outline four important practices for future research in membership inference to enable sound experimentation and inferences. In particular, (1) assessment for membership inference must be done in an IID setup where train and validation splits are from the same distribution, (2) experiments must be repeated over multiple random splits of the datasets, (3) experiments must be performed over multiple data distributions (4) careful experimentation must be done on both false positives and false negatives to ensure MIAs do not wrongly label non-members as members.

2.4 LLM Dataset Inference

Dataset inference builds on the idea of membership inference but attempts to leverage distributional properties in order to ascertain if a model was trained on a dataset or not. In our LLM dataset inference, we follow this recipe and use a combination of all the various MIA methods discussed in Section 2.2.1 for the data attribution in LLMs.

2.4.1 Procedure for the LLM Dataset Inference

We describe the procedure for our LLM dataset inference in four stages (see also visualization in Figure 2.1). Recall the initial example of a book writer who suspects that a portion of their books was trained on. We use this as a running example to describe the four stages of LLM dataset inference.

Stage 0: Victim approaches an LLM provider. A victim (author) $\mathcal{V}$ approaches an arbiter with a claim of ownership over data (book) that they suspect a model trainer or adversary $\mathcal{A}$ utilized. This stage involves the arbiter validating if the claim by $\mathcal{V}$ satisfies the assumptions under which dataset inference operates, that is, they provide an IID set of data that they suspect was trained on, and an equivalent dataset that $\mathcal{A}$ could not have seen, denoted as the validation set. This can, for instance, happen when authors have multiple drafts of a book chapter, and only one of the drafts makes it to the actual print. In such a case, $\mathcal{V}$ claims that $\mathcal{A}$ trained on the published version of their book, because $f(\mathcal{A})$ responds *differently* to the final versus rejected drafts of the book. Both suspect and validation sets are divided into non-overlapping splits (partitions) A and B. We will use A_{val}, A_{sus} in Stages 1, 2 and B_{val}, B_{sus} in Stage 3 to actually perform the ownership assessment.

Stage 1: Aggregate Features with MIAs. This step involves aggregating scores from various MIA methods described in the previous Sec. 2.2.1. A_{val}, A_{sus} sets are passed through the LLM under question to obtain their features derived from MIAs. Note that we use an aggregation of all the MIA methods discussed in the previous sections to create a single feature vector. In the next step, we will automatically determine which MIAs are useful for the task.

Stage 2: Learn MIA correlations. In this stage, we train a linear regressor to learn the importance of weights for different MIA attacks to use for the final dataset inference procedure. Across each MIA feature value, we first modify the top 5% outliers by changing their values to the mean of the distribution. This step is crucial to prevent issues in Step

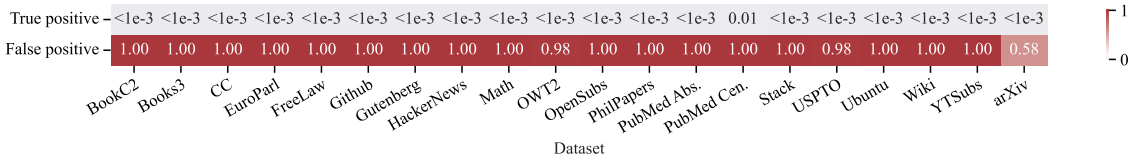


Figure 2.4: Result of dataset inference By applying dataset inference to Pythia-12b models with 1000 data points, we observe that we can correctly distinguish train and validation splits of the PILE with very low p-values (always below 0.1). Also, when considering false positives for comparing two validation subsets, we observe a p-value higher than 0.1 in all cases, indicating no false positives.

3, where the model might learn skewed correlations due to a few outlier samples. We then pass the data through a linear regression model to learn weights for each feature. All 'suspect' samples in A_{sus} are labeled as 0, and all validation samples in A_{val} are labeled as 1. A regressor is trained to predict the label given the samples, effectively learning the correlation between the features and their membership status.

Stage 3: Perform Dataset Inference. We use B splits of the suspect and validation sets, holding out up to 1000 samples in these splits for ownership assessment. Each sample is assigned a membership value using a trained linear classifier. These values are used to perform a statistical t-test to determine if the suspect set was used in training the model. We then report whether the model was trained on the suspect dataset based on the t-test results. For members of the suspect set, their confidence scores are significantly closer to 0 compared to non-members. The null hypothesis (H_0) is that the suspect dataset was not used for training. Assume that $\mu_{\mathcal{M}(B_{\text{sus}})}$ and $\mu_{\mathcal{M}(B_{\text{val}})}$ are the mean membership values of the suspect and validation sets, respectively. Then, H_0 and H_1 (alternate hypothesis) are:

$$H_0 : \mu_{\mathcal{M}(B_{\text{sus}})} \geq \mu_{\mathcal{M}(B_{\text{val}})}; \qquad H_1 : \mu_{\mathcal{M}(B_{\text{sus}})} \leq \mu_{\mathcal{M}(B_{\text{val}})}. \tag{2.1}$$

Combining p-values for Dependent Tests. In order to assess the significance of the results, we performed multiple t-tests using 10 different random seeds to obtain various splits of examples between A and B sets. Since the subsets had overlapping examples, the statistical tests are dependent [Vovk and Wang, 2020], and p-values must be aggregated accordingly [Brown, 1975, Kost and McDermott, 2002, Rüschendorf, 1982, Meng, 1994] . Let $p_1, p_2, \ldots, p_n$ denote the p-values obtained from the n t-tests performed with different random seeds. Under the null hypothesis, each p-value is uniformly distributed on the interval $[0, 1]$. We approximate the combined p-value by:

$$p_{\text{combined}} = 1 - \exp\left(\sum_{i=1}^{n} \log(1 - p_i)\right) \tag{2.2}$$

2.4.2 Assumptions for Dataset Inference

In order to operationalize dataset inference, we must obey certain assumptions on both the datasets (points 1 and 2 below), and the suspect language model (point 3 below).

1. The suspect train set and the unseen validation sets should be IID. This prevents the results from being confounded due to distribution shifts (such as temporal shifts in the case of WikiMIA).

2. We must ensure no leakage between the (suspected) train and (unseen) validation sets. The validation set should be strictly private, and only accessible to the victim.

3. We need access to the output loss of the suspect LLM in order to perform various MIAs.

2.4.3 Experimental Details

In our experiments, we aggregate 52 different Membership Inference Attacks (MIAs) in Stage 1 (many of which are overlapping and only differ in whether they capture the perplexity or the log-likelihood, or contrast the ratios or differences of model predictions). For the linear regression model trained in Stage 2, we train for 1000 updates over the data using simple weights over the 52 features. A total of 1000 examples are saved for training the regressor to learn correlations for stage 2, except in the false positive experiments where we use half the data. To aggregate scores from different MIAs, we **(i) normalize feature values** to ensure that all features aggregated across various membership inference attacks are on a comparable scale. Then, we **(ii) adjust values of outliers** before learning correlations with the classifier by replacing the top and bottom 2.5% of outlier values with the mean of that (normalized) feature. Finally, we **(iii) remove outliers before performing t-test** in Stage 3 once we have a single membership value from the regressor outputs for each sample in the B splits of the suspect and validation sets. Once again we remove the top and bottom 2.5% of outlier.

2.4.4 Analysis and Results with Dataset Inference

We analyze the performance of LLM dataset inference on the Pythia suite of models [Biderman et al., 2023a] trained on the Pile dataset [Gao et al., 2020]. We separately perform dataset inference on each and every subset of the PILE using the provided train and validation sets, and report the p-values for correctly identifying the training dataset. Before diving into various design choices, the key result is that dataset inference reliably finds the training distribution in all subsets of the PILE. (Figure 2.4). For the analysis of false positives, we carry out dataset inference on two splits of the validation set for each subset

of the PILE. Since neither of the validation subsets was used to train the model f, the returned p-values should be (and are) significantly above the selected threshold of 0.1 for any useful attribution framework. It is worth noting that the p-values for these tests are often remarkably low in the order of $1e-30$ and lower, suggesting high confidence in attributing dataset ownership. When contrasted with the lack of reliability of membership inference, dataset inference indeed shows great promise for future discourse on inspecting training datasets. We will now dive deeper into various ablations and results around dataset inference such as the features (membership inference attacks) chosen by the regressor, choice of pre-processing function, change in performance of dataset inference with model size, data duplication and number of permitted queries.

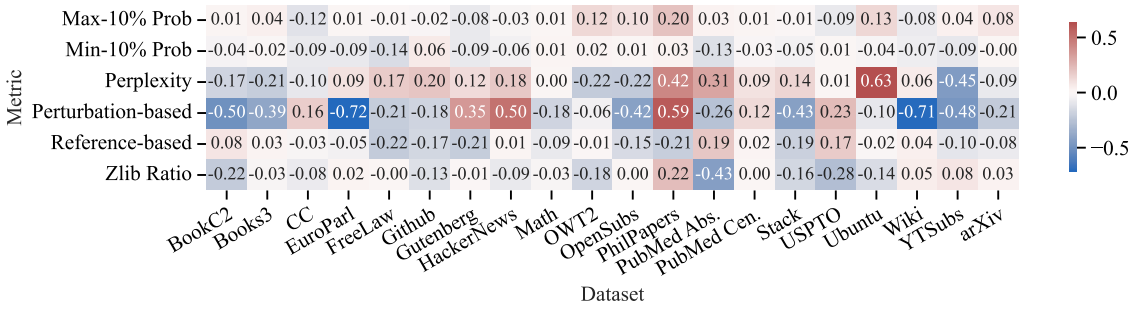
Feature Selection. For each domain, we find that the most prioritized metrics are different. This is the reason that the linear classifier is essential to appropriately determine the importance of each feature for determining per example membership, based on the dataset statistics. We present the results in Figure 2.5a. For example, while the Perturbation-based metric is necessary to be present for the CC dataset, it is not useful for the OWT2 dataset, which instead requires the Perplexity metric. Dataset inference automatically learns which MIAs are positively correlated with a given distribution. The linear regressor can be trained quickly on a CPU since it is only learning a weight assignment for each feature. Now, we investigate which MIAs get selected by dataset inference by analyzing the importance weights given by our linear regression to various MIAs.

Feature Pre-Processing. Considering the chosen features, we explore various pre-processing techniques to apply before training the linear regressor. The selected approach, referred to as Removal (Norm.) in Figure 2.5b, involves eliminating outliers and normalizing the feature values.

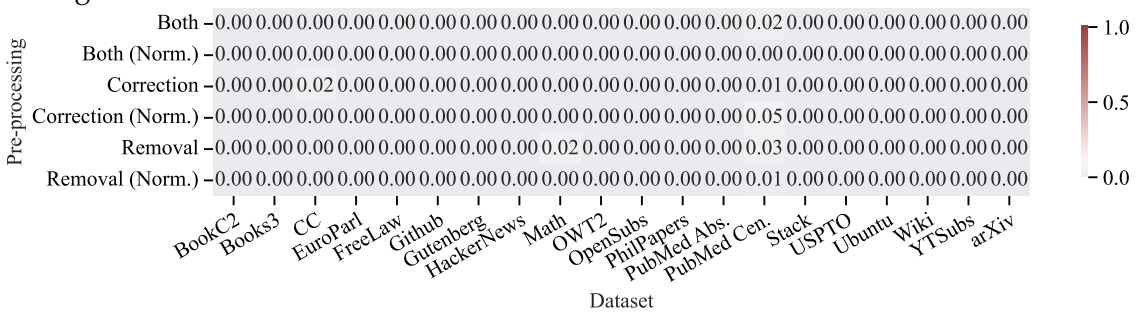
Number of Queries. We analyze the number of queries that have to be executed against the tested model f to determine if a given dataset was used for training. We present the results in Figure 2.6a. It can be seen more than half of datasets only require about 100 points, while 1000 points are sufficient to obtain p-values smaller than the threshold for all datasets.

Size of LLMs and Training Set Deduplication. By studying the Pythia suite of models [Biderman et al., 2023a] which are trained on the same dataset, we observe the success of dataset inference is positively correlated with the number of parameters in the LLMs. We present this result in Figure 2.6b as a violin plot to allow for visualizing distributions of

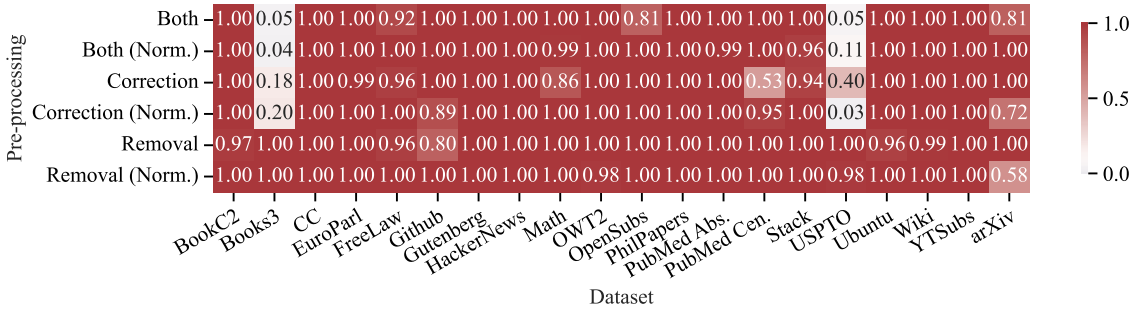
the datasets' p-values. It can be seen as the size of the model increases, the p-value distribution concentrates below the threshold of 0.1. This can be interpreted as the increase in the model memorization [Carlini et al., 2021b] due to the higher learning capacity, which provides a stronger signal for our dataset inference. We also contrast the models trained on deduplicated or non-deduplicated training sets when we are only allowed 500 query points. Observe that while the aforementioned trend holds for both kinds of models, the p-value distribution is more concentrated below 0.1 for the non-deduplicated models. This indicates memorization is more severe when some training data is duplicated, allowing various membership inference attacks to have a stronger signal.



(a) Feature Selection. A positive value indicates a correlation of the feature with the validation set, while a negative one denotes a correlation with the train set.

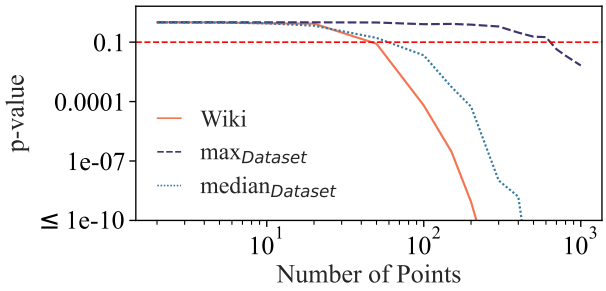


(b) Feature pre-processing. We present the p-values for a given dataset using differently pre-processed features.

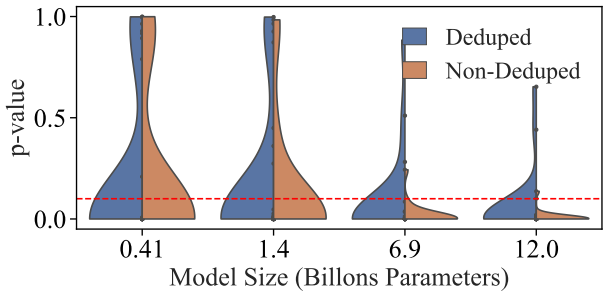


(c) Analysis of false-positives. A p-value smaller than 0.1 indicates a false positive.

Figure 2.5: Ablation study for dataset inference. We analyze which features based on or derived from the previous membership inference methods increase the success of dataset inference. (a) Our results indicate that no single feature contributes consistently, thus we need a linear model to selectively aggregate their impact on the final outputs from dataset inference. (b) Given the selected features, we consider different ways of how to pre-process them before building the classifier. The proposed method (denoted as Removal (Norm.)) removes outliers and normalizes the feature values. (c) We evaluate the selected and pre-processed features using suspect set that come from the validation data. We do not observe any false positives as shown in the last row in (c).



(a) Dataset inference success v.s. number of data points.



(b) Dataset inference success v.s. LLM size.

Figure 2.6: Ablation studies for the amount of data and model size. In (a), we plot the maximum and median p-values across all datasets, alongside the p-value of Wikipedia, as a function of the number of data points. In (b), a violin plot is made to show the distribution of p-values of the datasets with respect to the number of model parameters. Observe that dataset inference is more successful with more data and larger LLMs. It is also noteworthy that (a) dataset inference for a majority of datasets is accurate with less than 100 points, and (b) it is more accurate with respect to the non-deduplicated models that are trained on datasets with duplicated points. We hypothesize this is because the membership signal for most MIAs becomes stronger with the duplication of data.

Chapter 3

STAMP

Based on: STAMP
Saksham Rastogi*, Pratyush Maini*, Danish Pruthi. *Under review*

Abstract

Given how large parts of publicly available text are crawled to pretrain large language models (LLMs), data creators increasingly worry about the inclusion of their proprietary data for model training without attribution or licensing. Their concerns are also shared by benchmark curators whose test-sets might be compromised. In this paper, we present STAMP, a framework for detecting dataset membership—i.e., determining the inclusion of a dataset in the pretraining corpora of LLMs. Given an original piece of content, our proposal involves first generating multiple rephrases, each embedding a watermark with a unique secret key. One version is to be released publicly, while others are to be kept private. Subsequently, creators can compare model likelihoods between public and private versions using paired statistical tests to prove membership. We show that our framework can successfully detect contamination across four benchmarks which appear only once in the training data and constitute less than 0.001% of the total tokens, outperforming several contamination detection and dataset inference baselines. We verify that STAMPpreserves both the semantic meaning and utility of the original data. We apply STAMPto two real-world scenarios to confirm the inclusion of paper abstracts and blog articles in the pretraining corpora.

3.1 Introduction

To train large language models, much of the available text from the internet is crawled, allegedly including copyrighted material such as news articles and blogs [Grynbaum and Mac, 2023a,b]. Additionally, some evaluation datasets, originally intended for benchmarking model performance, may be compromised—an issue prominently discussed as *test-set contamination* [Magar and Schwartz, 2022, Jacovi et al., 2023, Sainz et al., 2023a]. A recent study reveals concerning evidence that pretraining corpora contain several key benchmarks [Elazar et al., 2024], and another demonstrates that impact of test set contamination

has been underestimated in many prominent LLM releases [Singh et al., 2024].

On one hand, training language models on copyrighted material might violate legal standards, and on the other, consuming test sets of machine learning benchmarks might offer a false sense of progress. Given the lack of regulations or incentives for model developers to disclose contents of their pretraining corpora [OpenAI, 2023, AI@Meta, 2024, Anthropic, 2024], it is critical to equip content creators with reliable tools to determine whether their content was included as a part of model training. Especially, third party approaches that can democratize detecting dataset membership and enable independent accountability.

Some approaches for detecting dataset membership embed random sequences in text or substitute characters with visually-similar unicodes [Wei et al., 2024]. However, such alterations impair machine readability, indexing and retrieval—making them impractical for content creators. More critically for benchmarks, such substitutions can alter tokenization, potentially compromising their utility for evaluation. Other proposals rely on access to a *validation* set that is unseen by the target model and drawn from the same distribution as the original dataset—a requirement hard to meet in practice [Maini et al., 2024c]. Recently, Oren et al. [2024] suggest comparing canonical ordering of test sets to random permutations, but this strategy assumes large portions of datasets are processed together within a single context window during pretraining. Most closely related to our proposal, Zhang et al. [2024a] use a statistical test to compare model confidence on original test instances and their rephrasings, assuming that the two distributions are identical—an assumption we show does not hold (Table 3.3).

In our work, we propose STAMP, a practical approach allowing creators to detect dataset membership through a statistical test with a probabilistic interpretation (Figure **??**). Our approach begins by taking the original content and generating multiple rephrased versions. Each rephrased version is watermarked using a distinct key for the hash function used in watermarking. Content creators can then release one of the generations publicly, while keeping the others private. A statistical test then evaluates the model likelihood of generating the public version against the private copies. For models that were trained on the publicly available generations, we expect to observe higher model likelihoods for these generations compared to their private counterparts.

Our work repurposes LLM watermarking to watermark documents that considerably enhance the detection sensitivity of our statistical test. (This is different from watermarking *models* themselves to prevent against model extraction attacks.) Specifically, we leverage the KGW watermarking scheme [Kirchenbauer et al., 2024], which embeds detectable signals by steering generations towards a randomly chosen "green" subset of the vocabulary.

We make all our code, data and models available at github.com/codeboy5/STAMP.

We empirically validate the effectiveness of our approach by continually pretraining the Pythia 1B model [Biderman et al., 2023b] on deliberately contaminated pretraining data. We contaminate the pretraining corpus by injecting test examples from four different benchmarks. Even with minimal contamination—that is, each test example appearing only once and each benchmark comprising less than 0.001% of the total training data—our approach significantly outperforms existing methods, achieving statistically significant p-values across all contaminated benchmarks. We also conduct a false positive analysis, wherein we apply our detection methodology to off-the-shelf pretrained LLMs that have not been exposed to the watermarked benchmarks and find that our test successfully denies their membership. Moreover, our analysis reveals that watermarking substantially enhances detection sensitivity, improving statistical significance by up to three orders of magnitude.

To demonstrate STAMP's effectiveness in detecting inclusion of copyrighted data in pretraining corpora, we present two expository case studies where we apply STAMPto detect membership of paper abstracts and blog articles. Our test achieves statistically significant p-values across these real-world scenarios. To further ensure that our framework preserves content quality, we conduct both automatic evaluations using GPT4 [OpenAI, 2023] and a human study, and find that STAMPmaintains content quality. These results highlight its utility in protecting copyrighted material (for creators), and detecting contamination (for auditors).

3.2 Related Works

3.2.1 Membership Inference Attacks (MIAs)

3.2.2 Test-set Contamination Detection

Prior works [Sainz et al., 2023a, Elazar et al., 2024] have demonstrated that common pretraining corpora are heavily contaminated with existing benchmarks. While some LLM providers [Brown et al., 2020c, Anil et al., 2023] report their decontamination protocols that are based on higher order n-gram analysis, most [Chowdhery et al., 2022, Jiang et al., 2023a, Anthropic, 2024] do not disclose any active decontamination measure. Post-hoc contamination analyses, which evaluate LLMs on uncontaminated test examples are more common, but often underestimate the impact of contamination, as shown by Singh et al. [2024]. While these analyses provide insight into contamination, they are limited to a narrow set of benchmarks and lack reproducibility preventing independent verification. Third-party tests to detect contamination will democratize contamination detection, enabling broader and more transparent evaluations without relying on model providers.

For third-party tests of benchmark contamination, Sainz et al. [2023b] propose to detect contamination by prompting the LLM to generate verbatim examples from the dataset. Similarly, Golchin and Surdeanu [2024] introduce *guided instruction*, which uses prompts containing dataset identifiers (e.g., dataset name and partition type) along with a random-length prefix, asking the LLM to complete the prefix. While effective, these heuristic approaches lack the ability to provide statistical proof of contamination or guarantees on the false positive rate. Additionally, they rely on strong memorization, requiring the LLM to regenerate instances closely resembling the original examples. Oren et al. [2024] propose a permutation test that compares the model's likelihood on the canonical ordering of a dataset (as it appears on the web) to a random shuffling of the same dataset. While this method provides statistical guarantees, it assumes that large portions of the dataset are seen together by the model during pretraining—a strong and often unrealistic assumption. PaCoST [Zhang et al., 2024a] involves constructing a rephrased counterpart for each original test example and detecting contamination through a statistical test that compares the model's confidence on the original and rephrased versions. We evaluate their approach in our experimental setup and find it to be ineffective.

3.2.3 LLM Memorization

Memorization in LLMs has primarily been explored in the context of training data extraction, with a extracting private information and examining the factors that influence the extent of memorization [Carlini et al., 2021c, Nasr et al., 2023a, Carlini et al., 2023b, Meeus et al., 2024a]. These studies have demonstrated that memorization tends to increase with (a) the size of the model, (b) the number of repetitions of the text, and (c) the perplexity of the text.

3.3 Preliminaries

In this section, we begin by formalizing the problem of detecting membership of a dataset (Sec. 3.3.1) and provide necessary background on watermarks for LLMs (Sec. 3.3.2).

3.3.1 Dataset Membership

The problem of dataset membership [Maini et al., 2021] aims to determine whether a dataset X has been included in the pretraining data D_{train} of a language model θ. We operate under a *gray-box* setting, where we can compute token probabilities for any sequence S but have no access to the pretraining data or model weights. Formally detecting membership of a dataset can be posed as a hypothesis test with the goal to distinguish between the following hypothesis:

- $\boldsymbol{H_0}$: θ is independent of X (no membership)
- $\boldsymbol{H_1}$: θ is dependent on X (membership),

where we treat θ as a random variable whose randomness arises from the sampling of the pretraining dataset D_{train} (which may or may not include X). Framing membership inference [Shokri et al., 2016] as hypothesis testing provides statistical guarantees on the false detection rate.

Our focus is on building statistical tests that can reliably detect dataset membership in language models. We aim to develop methods that make minimal assumptions about the format or nature of data—be it machine learning benchmarks, newsletters, or books.

3.3.2 Watermarks for LLMs

Watermarking techniques for LLMs embed subtle but distinctive patterns within generated text that are imperceptible to humans but algorithmically detectable. For our framework, we utilize the prominent KGW scheme [Kirchenbauer et al., 2024]. KGW scheme uses a

hash function that takes the context (preceding tokens) and a hash key h to partition the vocabulary V into two disjoint sets at each generation step: a green list G and a red list R.

To embed a watermark, the scheme biases the model's next-token probabilities by adding δ ($\delta > 0$) to the logits of tokens in the green list. Specifically, if $l_k^{(t)}$ denotes the original logit for token k at position t, then the modified logits are given by:

$$\hat{l}_k^{(t)} \leftarrow l_k^{(t)} + \delta \mathbb{1}\left[k \in G\right]. \tag{3.1}$$

3.4 STAMP: Spotting Training Artifacts through Watermarked Pairs

We introduce STAMP, a practical and principled framework that enables content creators to reliably detect whether their content was included in LLM pretraining data. Our approach builds on a key insight: if an LLM consistently prefers documents watermarked with a specific key (e.g., the key used for the publicly available version) over semantically equivalent content with distinct watermarks, then the model must have seen the preferred documents during pretraining. In this section, we detail how STAMPleverages this insight to create a robust statistical framework for membership detection. STAMPconsists of two stages: (1) a process for content creators to release watermarked content (Sec. 3.4.1) and (2) a paired statistical test to detect downstream dataset membership (Sec. 3.4.2).

3.4.1 Watermarking Datasets

The first stage of our approach involves generating multiple watermarked versions of a dataset through rephrasing. For a given dataset X, we employ an open-weights instruction-tuned LLM to generate rephrases. For each document q in the original dataset, we create a public version (denoted as $q^{'}$), where the rephrase is watermarked using a designated public key as the *hash key*. Additionally, we generate m private versions (denoted as $q_1^{''}, q_2^{''}, \ldots, q_m^{''}$), where each generation is watermarked using a distinct private key as the hash key. The public version is released online, while the private versions are kept confidential. Crucially, due to the design of our test relying on pairwise comparisons at a document level (Sec. 3.4.2), each document q in a dataset X can use a different set of hash keys. This ensures that introducing watermarking during the rephrasing stage does not alter the token distribution of the dataset X and, importantly, preserves the overall token distribution of the internet data.

LLM Watermarks as Sampled Markers. While watermarking is traditionally intended for attributing generated text to a specific LLM, our motivation diverges from this original purpose. First, we leverage LLM watermarking as a mechanism to embed distinct signals into the rephrases through the use of distinct hash keys. The randomness in both our hash key selection and the watermarking process itself enables us to frame the detection problem as hypothesis testing. Under the null hypothesis $\boldsymbol{H_0}$ (no membership), the target model shouldn't favor content watermarked with any particular key. Second, the watermarking process itself introduces subtle perturbations that increase sequence perplexity, which has been empirically shown to enhance memorization during training [Meeus et al., 2024a], further amplifying our ability to detect membership.

3.4.2 Detecting Dataset Membership

To detect membership, we leverage the insight that under the null hypothesis $\boldsymbol{H_0}$ (no membership), the model should not exhibit any systematic preference towards any of the semantically equivalent paraphrases of documents that are watermarked with distinct keys—the public version of the dataset and privately held versions of the dataset. This follows from the randomness inherent in our selection of keys and nature of watermark we employ. We formalize this intuition through a statistical testing framework.

For each document q, we compute the perplexity difference d_i between its public version $q_i^{'}$ and private version $q_i^{''}$ that form a pair $(q_i^{'}, q_i^{''})$:

$$d_i = PPL_\theta(q_i^{'}) - PPL_\theta(q_i^{''}). \tag{3.2}$$

Prior to applying the paired t-test, we modify the top 5% outliers by clipping their values. This prevents issues where the test can become ineffective due to a few outlier samples. Under the alternative hypothesis $\boldsymbol{H_1}$, we expect these differences to be negative on average, indicating lower perplexity for public versions. We evaluate this using a one-sided paired t-test statistic:

$$t = \frac{\bar{d}}{s_d/\sqrt{n}}, \tag{3.3}$$

where $\bar{d}$ and s_d are the mean and standard deviation of the differences across the collection of documents respectively and n is the number of documents [Student, 1908]. The one-sided p-value specifically tests for $\bar{d} < 0$, following our alternative hypothesis that exposure during training leads to lower perplexity on public versions. Paired tests provide higher statistical power, enabling detecting membership even with a smaller collection of documents (n), as we show in our experiments.

Multiple Private Keys. In practice, we empirically observe that models may exhibit inherent biases at an individual document level, occasionally assigning lower perplexity to specific private rephrases (q'') independent of membership. To make our detection robust against such biases, we propose using multiple private rephrases, each watermarked with a distinct key. Instead of comparing against a single private version, we test whether public rephrases exhibit lower perplexity compared to the average perplexity across m different private rephrases:

$$d_i = PPL_\theta(q_i^{'}) - \frac{1}{m}\sum_{j=1}^{j=m} PPL_\theta(q_{i,j}^{''}), \tag{3.4}$$

where m is a hyperparameter known as *private key count*, and $q_{i,j}^{''}$ represents the j^{th} private rephrase of i^{th} document. Through controlled experiments, we analyze the effect of this hyperparameter on statistical strength of our test (Sec. 3.5.4).

3.5 Experiments & Results

Dataset (↑)	Pythia 1B	Gemma-2 2B	Mistral 7B	LLaMA-3 8B	Gemma-2 9B
TriviaQA	0.52	0.91	0.94	0.65	0.91
ARC-C	0.31	0.25	0.12	0.26	0.37
MMLU	0.54	0.41	0.29	0.24	0.43
GSM8k	0.38	0.16	0.26	0.71	0.37
Paper Abstracts	0.55	0.74	0.83	0.63	0.89
Blog Articles	0.21	0.72	0.74	0.88	0.12

Table 3.1: False positive analysis on off-the-shelf LLMs. We apply STAMPon LLMs that have not seen the datasets and report the p-values. Our results (high p-values) show that our method is robust against false positives.

To evaluate the ability of STAMPfor membership detection, we first focus on benchmark contamination—the inclusion of evaluation benchmarks in the pretraining corpora of LLMs. This setting presents unique challenges for membership detection. First, benchmarks must maintain their utility as reliable indicators of progress, which constrains the modifications we can make prior to their release. Second, benchmarks typically contain limited text compared to other content types (e.g., books or newsletters), making detection particularly challenging.

Dataset	(↑) Pythia Uncontaminated	(↑) Pythia Contaminated
TriviaQA	0.52	0.28
ARC-C	0.31	0.56
MMLU	0.54	0.15
GSM8K	0.38	0.47
Abstracts	0.55	0.07
Blogs	0.21	0.73

Table 3.2: False positive analysis. *Pythia Uncontaminated* denotes the p-values on a pretrained Pythia model that has not been contaminated. *Pythia Contaminated* refers to p-values when testing for membership of *held-out* subsets of datasets on a model contaminated with different subsets of the same datasets. High p-values denote that our approach does not falsely detect membership.

Dataset	Original vs Rephrased	Public vs Private
TriviaQA	0.66	0.51
Arc-C	0.83	0.52
MMLU	0.83	0.53
GSM8k	0.84	0.57
paper abstracts	0.86	0.57

Table 3.3: AUROC using *bag-of-words* features to distinguish between different versions of datasets. The first column shows AUROC for distinguishing original datasets from their rephrased versions, where high values (> 0.8) indicate clear distributional differences. The second column shows AUROC for distinguishing between *public* and *private* watermarked versions, where values near 0.5 indicate distributional similarity.

3.5.1 Releasing Watermarked Test Sets

We evaluate our approach using four widely-used benchmarks: TriviaQA [Joshi et al., 2017], ARC-C [Clark et al., 2018a], MMLU [Hendrycks et al., 2021a], and GSM8K [Cobbe et al., 2021b]. For each benchmark, we follow our proposed methodology (Sec. 3.4.1) to generate watermarked public and private paraphrases. We use the instruction tuned Llama3-70B [AI@Meta, 2024] model and a benchmark-agnostic prompt (provided in Appendix B) to generate these rephrased copies. For each benchmark, we randomly select one watermarked version to be the *public* version. Examples of the rephrased test instances are provided in Appendix B.

Key Distinction. While rephrasing has been previously explored for detecting contamination [Zhang et al., 2024a], existing approaches typically compare human-written content against their LLM-generated rephrases, overlooking a crucial confounding factor: language models exhibit systematic preferences for LLM-generated text over human-written content [Liu et al., 2023d, Mishra et al., 2023, Laurito et al., 2024]. This inherent bias undermines the reliability of statistical approaches that compare human-written content with their LLM rephrasings, as any detected differences might stem from this general preference rather than training exposure. To enable reliable statistical testing, it is crucial to control the data generating process for both versions being compared. We address this by ensuring both our public and private versions are generated through the same process, differing only in their watermarking keys. Given the random selection of keys, we expect no systematic preferences between versions unless one was seen during training.

We empirically validate that human-written content and its LLM-generated rephrasings are easily distinguishable (thus violating the expected IID requirement): a simple bag-of-words classifier obtains AUROC > 0.8 on four out of five benchmarks, whereas the classifier performs no better than random chance when distinguishing between rephrasings watermarked with different keys. Detailed analysis and classifier specifications are provided in Appendix B.

3.5.2 Pretraining with Intentional Contamination

Setup. To simulate downstream benchmark contamination as it occurs in real-world scenarios and evaluate the effectiveness of our test, we perform continual pretraining on the 1 billion parameter Pythia model [Biderman et al., 2023b] using an intentionally contaminated pretraining corpus. The corpus is a combination of OpenWebText [Contributors, 2023] and *public* watermarked version of the four benchmarks, as mentioned in Section 3.5.1. Each test set accounts for less than **0.001%** of the pretraining corpus, with exact sizes detailed in Table 3.7 in the appendix. All test sets in our experiments have a duplication rate of 1 (denoting no duplication whatsoever), and the overall pretraining dataset comprises 6.7 billion tokens. Details of the exact training hyperparameters are provided in Appendix B.

Baselines. We compare STAMPagainst two recent statistical approaches to detect membership: PaCoST [Zhang et al., 2024a] and LLM DI [Maini et al., 2024c]. PaCoST employs a paired t-test that compares model confidence on original and rephrased versions, while LLM DI aggregates multiple membership inference attacks (MIAs) to perform statistical testing. For LLM DI, which requires access to an unseen *validation* set, we evaluate two settings: (1) using private rephrases of the publicly available dataset as the *validation* set, and (2)

using an entirely different set of documents from the same distribution as the *validation* set.

Additionally, we also evaluate state-of-the-art MIAs: *PPL* [Yeom et al., 2018b], *Zlib* [Carlini et al., 2021c], *Min-K* [Shi et al., 2024b], *Min-K++* [Zhang et al., 2024b] and *DC-PDD* [Zhang et al., 2024e]. Since MIAs rely on a non-trivial detection threshold, we report AUROC scores across two settings: (1) discriminating between public rephrases in training and private rephrases of the same documents, and (2) discriminating between public rephrases in training and unseen documents from the same dataset.

Main Results.

False Positive Analysis. To ensure the robustness of STAMPagainst false positives, we conduct two key experiments. First, we apply our detection methodology to off-the-shelf pretrained LLMs that have not been exposed to the watermarked benchmarks. The results for Pythia 1B, presented in the first column of Table 3.2, show no false positives. We extend this analysis to models of different sizes and families in Table 3.1, consistently finding no false positives across all tested models, confirming the robustness of STAMPagainst false positives. Second, we perform a stronger test to evaluate whether STAMPdetects the membership of the dataset rather than just distributional differences due to different watermarking keys. We create held-out subsets from the same benchmarks and watermark them using the identical public keys used for our contaminated versions. While these held-out sets share the same distribution and watermarking as our training data, they contain entirely different examples. We then apply our detection methodology to test if these *held-out* sets are falsely detected as members in our contaminated Pythia 1B model. The second column of Table 3.2 shows consistently large p-values, indicating STAMPsuccessfully refutes membership for these *held-out* sets.

Performance Without Watermarks Embedded. To validate our hypothesis that using a watermarked LLM to generate the rephrased copies of the benchmark enhances the statistical strength of our test, we conduct experiments under the same settings as described above (Sec. 3.5.2), but with rephrased copies generated without using a watermarked LLM. The results, presented in Table 3.4, confirm that incorporating watermarked test sets significantly boosts the statistical power of our test, improving performance by at least two orders of magnitude across all benchmarks.

Method	Benchmark (↓)			
	TriviaQA	Arc-C	MMLU	GSM8K
PaCoST [Zhang et al., 2024a]	**1.6e-3**	0.33	0.19	0.21
LLM DI [Maini et al., 2024c] (*same*)	0.43	0.31	0.46	0.30
LLM DI [Maini et al., 2024c] (*different*)	**0.02**	0.53	**0.03**	0.71
STAMP(w/o paired tests)	0.14	0.07	0.08	**0.02**
STAMP(w/o watermarking)	**0.02**	**5.1e-3**	**0.02**	**1.4e-3**
STAMP	**1.2e-4**	**2.8e-4**	**7.0e-4**	**6.6e-6**

Table 3.4: P-values for detecting *test-set contamination* for different methods. For LLM DI [Maini et al., 2024c], *same* refers to using rephrases of the benchmark questions as *validation* set, while *different* uses an entirely different set of unseen questions from the same benchmark as the *validation* set. **Bold** indicates statistically significant results ($p < 0.05$). Across all the four benchmarks, our approach results in lower p-values compared to other approaches (lower is better).

3.5.3 Utility of Test Sets

Detecting contamination alone is insufficient; the watermarked content should retain the desired properties (for e.g., benchmarks should maintain their utility as reliable indicators of LLM performance). Using the lm-evaluation-harness framework [Gao et al., 2024c], we assess five pre-trained LLMs on both original and watermarked benchmarks. Additionally, we measure semantic preservation using the P-SP metric [Wieting et al., 2022].

Our results, presented in Table 3.5, demonstrate that STAMP-watermarked variants maintain benchmark utility: LLMs achieve similar absolute performance and the relative rankings of LLMs across all benchmarks are unaffected. In contrast, unicode watermark [Wei et al., 2024] significantly degrades benchmark utility, with performance drops of up to 20% and does not preserve relative rankings. STAMP-watermarked variants also result in high semantic preservation (P-SP scores between 0.83 & 0.91) across all benchmarks. For reference, the average score of human paraphrases is 0.76 as per [Krishna et al., 2023].

These results are available in Table 3.6.

3.5.4 Parameters Affecting the Power of the Test

Benchmark size. To analyze the effect of sample size (n) on detection power, we evaluate our test on benchmark subsets ranging from 100 to 1000 examples. For each size, we average p-values across 10 runs with different random seeds. Our results, in Figure **??**, demonstrate that our approach works even with just 600 examples, where we consistently

Dataset	Metric	Variant	Pythia 1B	Gemma-2 2B	Mistral 7B	LLaMA-3 8B	Gemma-2 9B
ARC-C	0-shot	Original	26.1	48	49.1	50.6	59.0
		Unicode	21.6	37.3	39.0	41.5	49.8
		STAMP	26.3	46.8	49.1	50.5	57.1
MMLU	5-shot	Original	28.1	52.9	59	61.1	68.6
		Unicode	28.4	45.0	51.5	55.9	63.2
		STAMP	28.8	51.6	56	61.8	68.4
TriviaQA	5-shot	Original	12.4	52.7	67.2	68.9	70.1
		Unicode	1.1	23.6	46.0	44.3	54.8
		STAMP	11.4	51.9	65.9	66.3	68.6
GSM8K	5-shot	Original	1.6	25.8	34.4	51.8	65.5
		Unicode	1.5	23.1	23.3	46.7	60.8
		STAMP	2.2	27.2	37.5	54.9	65.8

Table 3.5: Performance of models on the original datasets compared to the watermarked benchmarks. We evaluate the models using the LM evaluation harness [Gao et al., 2024c] with the default settings, comparing performance on original benchmarks against two watermarking approaches: Unicode substitutions [Wei et al., 2024] and STAMP. We find that models obtain comparable performance on STAMP-watermarked benchmarks, but crucially, **the relative ranking of LLMs remains unchanged across all benchmarks**, demonstrating the utility of watermarked benchmarks in comparing models.

achieve low p-values ($\approx 10^{-3}$) across all datasets.

Private key count. Our proposed test compares the perplexity of the public version against the average perplexity of m private versions (Equation 3.4). Here we analyze how this hyperparameter (m) affects the statistical power of our test. As shown in Figure 3, increasing the number of private keys strengthens detection up to a threshold of 5 keys, beyond which we see negligible improvement.

Size of Pretraining Corpora. We analyze our test's effectiveness for different scales of pretraining data by combining contaminated benchmarks with varying amounts of OpenWebText data [Contributors, 2023]. We note that while the strength decreases with corpus size, the *rate of decline* diminishes substantially beyond 4 billion tokens, with minimal drop in detection strength between 4 and 6 billion tokens (Figure 3.1). Notably, these results are obtained with a modest 1B-parameter model; given that larger models exhibit stronger memorization [Carlini et al., 2019c], we believe that STAMPwill detect membership for

	Triv-QA	Arc-C	MMLU	GSM8K	Abs.
P-SP	0.91	0.83	0.86	0.90	0.95

Table 3.6: Semantic similarity scores (P-SP) [Wieting et al., 2022] between original datasets and their watermarked rephrases (higher is better). Triv-QA and Abs. refers to TriviQA and paper abstracts respectively. For reference: the P-SP value is 0.76 for human-written paraphrases as per a recent study [Krishna et al., 2023].

Benchmark	Size (Tokens)	% Pretraining Data
TriviaQA	34609	5.1e-4
Arc-C	36863	5.5e-4
MMLU	42548	6.3e-4
GSM8k	61132	9.0e-4

Table 3.7: Size of evaluation benchmark used in the intentional contamination experiment (Sec. 3.5.4). Each benchmark is subsampled to 1,000 examples, with each injected benchmark making up less than 0.001% of the entire pretraining corpus, which consists of 6.7 billion tokens. Each benchmark is injected exactly once into the corpus without any duplication.

larger models.

3.6 Real World Case Studies

To demonstrate STAMP's effectiveness in detecting *unlicensed* use of copyrighted data in model training, we present two expository case studies. Specifically, we apply STAMPto detect membership of (1) abstracts from EMNLP 2024 proceedings [emn, 2001] and (2) articles from the AI Snake Oil newsletter [Narayanan and Kapoor, 2023].

Paper Abstracts. We sample 500 papers from EMNLP 2024 proceedings [emn, 2001] and generate watermarked rephrasings of their abstracts. Additionally, we generate watermarked rephrasings for another set of 500 abstracts, which we use as a *held-out* validation set for our experiments. The prompt templates used for rephrasing and examples of watermarked abstracts are provided in Appendix B and Appendix B , respectively.

To evaluate whether the semantic content of abstracts is preserved, we use the P-SP metric [Wieting et al., 2022], where watermarked abstracts achieve a high score of 0.95, indicating that the semantic content is largely preserved. To further evaluate the acceptability of watermarked abstracts, we conduct both an automated evaluation (using GPT-4) and a small-scale human study involving original authors. In both evaluations, the

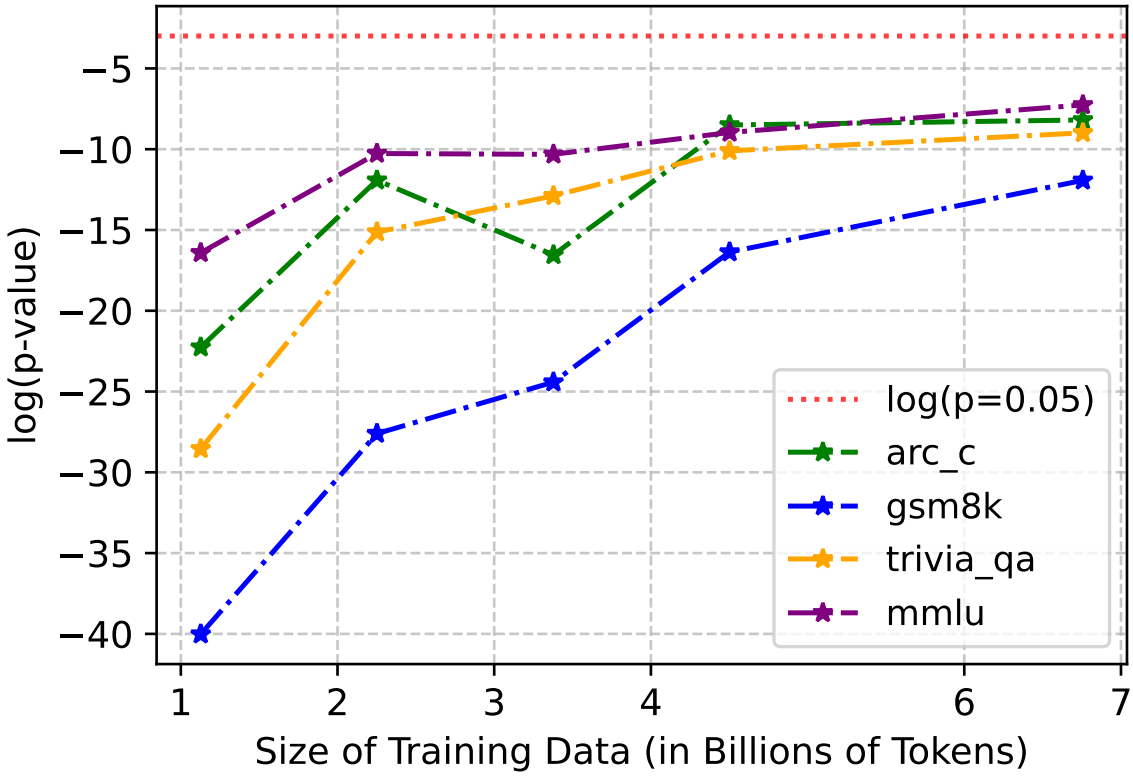


Figure 3.1: Log p-value vs pretraining corpus size. We observe that the *rate of decline* diminishes as we increase the corpus size, with negligible drop between 4B and 6B.

original abstract and its watermarked rephrasing is compared, classifying the rephrased content into one of five options: *preferred*, *acceptable*, acceptable with *minor revisions*, or *major revisions*, and lastly *unacceptable*. Further details are available in Appendix B.

For 1000 watermarked abstracts, 99% were rated by GPT-4 as either *preferred* or *acceptable*. In a preliminary human study, we ask authors to review rephrasings of their own abstracts. Our results in Figure 3.2 show that out of the 40 watermarked abstracts evaluated, authors find 24 to be acceptable as is, indicate 11 could use minor edits, and 4 prefer the rephrased version over their self-written abstracts, with just 1 abstract requiring major edits. Details of the evaluation are provided in Appendix B.

Blog Posts from AI Newsletter. We collect 56 posts from the popular AI Snake Oil newsletter [Narayanan and Kapoor, 2023], and use 44 for pretraining and hold 12 for validation. To demonstrate how STAMPcould handle longer-form content, we adapt it to rephrase at the paragraph level, treating each paragraph as an independent datapoint for our test. Note, while each paragraph serves as a datapoint for our test, the blog posts are

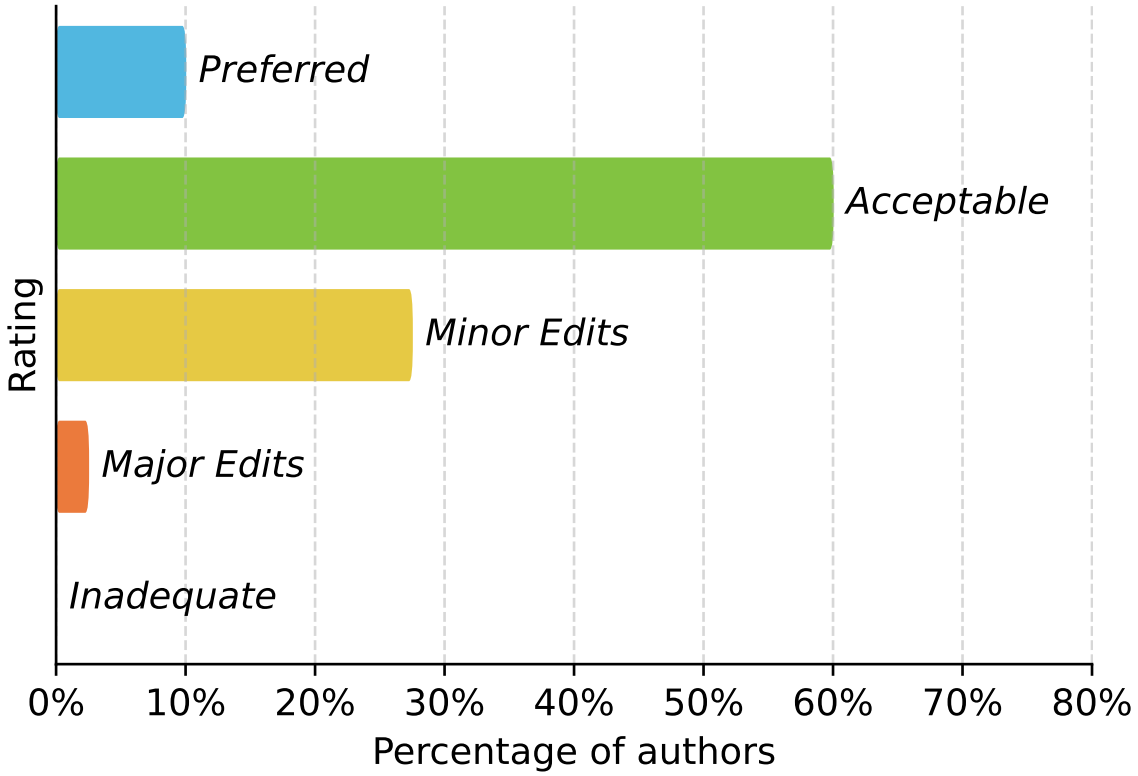


Figure 3.2: Author evaluation of watermarked abstracts. Out of the 40 total abstracts, for 4 of them the watermarked version was *preferred*, for another 24 it was found to be *acceptable*, 11 required *minor edits* with just 1 requiring *major edits*, suggesting that watermarking preserve quality.

included in the pretraining corpora at the document level, following standard pretraining practices (detailed in Appendix B). We present the prompt used to rephrase in Appendix B

We evaluate STAMP's ability to detect dataset membership by performing continual pretraining on the Pythia 1B model using a training corpus composed of watermarked paper abstracts (≈ 105K tokens), watermarked blog posts (≈ 95K tokens), and a subset of OpenWebText (≈ 3.3B tokens). Additionally, to verify that STAMPcan detect dataset membership for distinct datasets watermarked with the same key, we apply a consistent watermarking key when generating the public versions of both datasets.

Results. Our results in Table 3.8 demonstrate that STAMPeffectively detects dataset membership for both paper abstracts and blog posts, achieving statistically significant p-values. To compare, we evaluate LLM DI under different choices of validation set: first, using private rephrases of the same documents and second, using a different held-out set of documents watermarked with the same public key. While LLM DI can detect membership

Method (↓)	Paper Abstracts	Blog Articles
LLM DI (same)	0.15	0.44
LLM DI (different)	**0.05**	0.58
STAMP (w/o paired tests)	**0.01**	0.07
STAMP	**2.7e-12**	**2.4e-3**

Table 3.8: Case studies. We report p-values of different approaches for detecting dataset membership (lower is better). LLM DI (same) uses the private rephrasing of the same documents, while LLM DI (different) uses different documents from a held out set from the same distribution.

for paper abstracts, it fails to do so for blog posts. Further, membership inference attacks exhibit near-random performance (Table 3). To verify the robustness of STAMPagainst false positives, we evaluate it under the two settings discussed earlier (Sec. 3.5.2). Our results in Table 3.2 confirm that STAMPdoes not result in any false positives, reinforcing its reliability.

3.7 Discussion

In this work, we demonstrate how LLM watermarks offer a viable option for creators to watermark their content prior to release, enabling principled detection of dataset membership. Our choice of watermarks offers the key benefit of controlling distortion strength, allowing us to balance detectability with content preservation. Specifically, we show that with a modest signal strength ($\delta = 1.0$ for KGW watermarking scheme), we can successfully detect dataset membership while maintaining content fidelity. Future work could explore optimal watermark strengths and designs for different data distributions. Particularly promising is the detection of code dataset membership, as our watermarking approach preserves machine readability. This opens up new possibilities for tracking code inclusion in model training.

Limitations. There a few important limitations of our work to consider. First, our method requires embedding watermarks prior to content release and cannot be retroactively applied to existing published content.

Second, while our method does not require access to the pretraining corpora or model weights, it does require access to token probabilities from the model (gray box access). While this is not a concern for open weights models [Liang, 2024], having appropriate legal frameworks could enable our method to be applied to LLMs with restricted APIs.

Third, due to computational constraints, we evaluate our approach using continual pretraining rather than training models from scratch. While our results demonstrate effectiveness in this setting outperforming baselines, future work could validate these findings with models trained from scratch.

Maybe it'll help to add here that we are not considering that the developers are malicious, who may re-run the content through another LLM first. can add but with our broader focus on copyright it should also be okay to skip, i feel..

3.8 Conclusion & Future Directions

In this work, we presented STAMP, a statistical framework for detecting dataset membership, which can reliably be used by content creators to watermark their content, while preserving the utility, or the meaning, of the original content. We demonstrated STAMP's effectiveness in detecting test-set contamination through comprehensive experiments. Our ablation studies systematically analyzed how detection strength varies with dataset size, the number of private versions, and pretraining corpus size. We validated the real-world applicability of our approach through two case studies: detecting paper abstracts and blog posts in pretraining data.

There are several **important limitations** of our work: first, watermarks must be embedded before the content is released online, making it inapplicable to already published content. We believe this is a fundamental limitation shared by existing statistical methods, as they require knowledge of the data-generating process to construct a valid null distribution. Second, our method requires access to token probabilities from the model (gray box access). Third, while our human study showed that majority of authors found the rephrasings to be acceptable, rephrasing could introduce errors in the content. However, we believe this will be less of a concern moving forward as general model capabilities, including paraphrasing quality, continue to improve. Finally, due to computational constraints, we evaluated our approach using continual pretraining rather than training models from scratch. While our results demonstrate effectiveness in this setting outperforming baselines, future work could validate these findings using models that are trained from scratch.

Future work could explore the optimal watermarking strength for different data distributions (and use cases) to balance a (plausible) trade-off between detectability and quality of watermarked content. Future work could also validate, or extend, our approach to other domains, such as code, speech, images or videos.

Part III

Mechanisms and Localization of Memorization

Overview

Memorization is not monolithic: where it lives in a network, how it emerges during training, and how it manifests at inference all vary with data, architecture, and objective. This Part dissects memorization at three levels. First, we localize memorization within networks, showing it often concentrates in small neuron subsets rather than only in late layers. Second, we propose adversarial compression views—text for LLMs and image conditioning for diffusion models—that operationalize when a model effectively "stores" examples. Third, we examine mechanisms that promote controllable memorization, enabling isolation that can later be removed with minimal collateral damage.

By connecting mechanistic insights to measurable definitions, these chapters clarify when memorization is benign, when it is risky, and how to interrogate it reliably in both language and vision generators.

Guiding questions

- Q1: Can we pinpoint which layers/neurons are critical for memorization, and how small can that set be?
- Q2: When is a string or image "memorized" under adversarial compression, and how robust is this criterion to prompt/style changes?
- Q3: What training-time mechanisms encourage memorization to be isolated rather than entangled with general ability?
- Q4: How do these insights generalize across architectures (transformers, diffusion) and datasets?

Chapter 4

Can Neural Network Memorization Be Localized?

Based on: **?**: Pratyush Maini, Michael C. Mozer, Hanie Sedghi, Zachary C. Lipton, J. Zico Kolter, Chiyuan Zhang. *Can Neural Network Memorization Be Localized?*. ICML 2023

Abstract

Recent efforts at explaining the interplay of memorization and generalization in deep overparametrized networks have posited that neural networks *memorize* "hard" examples in the final few layers of the model. Memorization refers to the ability to correctly predict on *atypical* examples of the training set. In this work, we show that rather than being confined to individual layers, memorization is a phenomenon confined to a small set of neurons in various layers of the model. First, via three experimental sources of converging evidence, we find that most layers are redundant for the memorization of examples and the layers that contribute to example memorization are, in general, not the final layers. The three sources are *gradient accounting* (measuring the contribution to the gradient norms from memorized and clean examples), *layer rewinding* (replacing specific model weights of a converged model with previous training checkpoints), and *retraining* (training rewound layers only on clean examples). Second, we ask a more generic question: can memorization be localized *anywhere* in a model? We discover that memorization is often confined to a small number of neurons or channels (around 5) of the model. Based on these insights we propose a new form of dropout—*example-tied dropout* that enables us to direct the memorization of examples to an a priori determined set of neurons. By dropping out these neurons, we are able to reduce the accuracy on memorized examples from 100% to 3%, while also reducing the generalization gap.

4.1 Introduction

Deep neural networks are capable of memorizing randomly labeled data [Zhang et al., 2017] yet generalize remarkably well on real world data. Theories have been developed to understand this puzzling phenomenon. For example, *implicit regularization* argues that the

training algorithms, i.e., SGD and its variants, implicitly encourage the trained model to converge to generalizable solutions when the underlying data distribution is well-behaved [Soudry et al., 2018, Smith et al., 2021]. Alternatively, the theory of *benign overfitting* argues that when a neural network is sufficiently overparameterized, it generalizes to the majority of the dataset using a *simple* component of the input, and memorizes mislabeled and irregular data using a *complex* component of the input data [Bartlett et al., 2020]. These components do not interfere, making such overfitting benign. Another theory of label memorization suggests that when the data follows a long-tailed distribution, memorizing rare examples in the tail is necessary for achieving optimal generalization [Feldman, 2020b, Feldman and Zhang, 2020]. One common notion of memorization is the 'leave-one-out' memorization—does the prediction of an (atypical) example change significantly if it was removed from the training set? Since this is computationally expensive, multiple prior works operationalize memorization via mislabeled examples, which *must* be memorized for correct prediction. In our study, we consider both types—mislabeled and atypical example memorization.

The practical implication of memorization in deep neural networks has also been widely studied, especially from a privacy and security perspective. For example, Carlini et al. [2021d] shows that it is possible to extract training data from large language models. Other work has attempted to reconstruct training examples from output activations [Teterwak et al., 2021] or even parameters [Haim et al., 2022] of *discriminatively* trained neural networks. However, previous studies have largely focused on the implications of memorization and/or identifying memorized examples. The question of *where memorization happens* in a neural network remains unclear with only anecdotal observations. In particular, it is *commonly* believed that memorization of training examples happens in the last (or final few) layers of a deep network [Baldock et al., 2021, Stephenson et al., 2021].

In this work, we show that memorization is confined to a small subset of neurons distributed across all layers, as opposed to a fixed set of (such as the final few) layers. We conduct a systematic study to understand the mechanism of memorization in neural networks, in particular, whether there are specific locations in which this phenomenon materializes. Then, we propose a simple dropout modification that enables directing memorization to a fixed set of network neurons. We perform our investigation in two stages.

In Stage I, to verify previous beliefs that memorization is confined to the last (few) layers, we design two experiments to detect memorization at the layer level. First, by measuring the influence of memorized (noisy) examples on model weights via gradient accounting (Section 4.4) we make two key discoveries: (i) the average contribution to gradient norms by memorized examples is an order of magnitude larger than that by typical (clean) examples; and (ii) gradient contribution from the two groups of examples (clean and noisy) have an

extremely negative cosine similarity throughout training, at every layer of the model. This suggests that at a layer level, the learning of noisy examples opposes the learning of clean examples, and indicates that mislabeled example overfitting may not be benign in practical settings. Interestingly, the last layer, or any other individual layer, does not stand out as receiving more influence from the memorized examples than from the typical examples. Second, to understand the functional influence of these weight updates, we draw inferences via weight rewinding and retraining (Section 4.5). In *weight rewinding*, we replace weights of individual layers of the converged model with earlier training checkpoints, and measure the accuracy on the training set. We find that the final layers are not critical to the correct prediction of noisy examples. Moreover, as the model and problem complexity change, we find that the critical layers of the model begin to change, suggesting the critical layers, if any, may be dependent on the particular problem instance. To build on the experiment of layer rewinding, keeping the rest of the model frozen, we retrain the re-initialized layers only on clean examples, and make a surprising discovery—for most layers (including the last few layers), training on clean examples confers nearly 100% accuracy on the (now unseen) mislabeled examples—i.e., the model predicts the randomly assigned (incorrect) label even without seeing that example, indicating the redundance of these layers for memorization.

In Stage II, we move away from the hypothesis of layer-level example memorization and inspect memorization at a neuron level, akin to the "grandmother cell" of Barlow [1972] (Section 4.6). By iteratively removing important neurons with respect to a sample, we determine the minimum number of neurons required to flip an example's predictions. We find that memorized (noisy) examples on average require significantly fewer neurons (by a factor of 1/3) for flipping the predictions, as opposed to clean examples. Moreover, we make the observation that the neurons chosen as most important by our greedy search algorithm are scattered over multiple layers of the model, and the distribution of such neurons for both clean and mislabeled examples are similar.

Inspired by this observation, we develop a training mechanism that enables us to intentionally direct the memorization of examples to a predetermined set of neurons. Specifically, we propose a new form of dropout that we call *example-tied dropout* (Section 4.6.2). We keep a fraction of (generalization) neurons always active during training. A complimentary small fraction of example-specific (memorization) neurons are only activated when that example is sampled in a mini-batch. Using this training scheme, we are able to strongly concentrate the memorization of examples to the memorization neurons. In particular, dropping out these neurons during evaluation brings down the accuracy on mislabeled examples from 100% to 0.1% in the case of the MNIST dataset and from 99% to 3% on the CIFAR10 dataset, with a minor impact on the accuracy on clean training exam-

ples. Moreover, we note that the clean examples that get misclassified upon dropping out of the memorization neurons are in fact atypical or ambiguous examples.

In summary, our findings suggest that memorization of examples by neural networks can not be localized to the last, or any few layers, but is often determined by a small set of neurons that may be distributed across layers. Moreover, we can perform training time modifications to restrict the model's memorization to a predefined set of neurons.

4.2 Related Work

Learning mechanism. The learning dynamics of neural networks and their interplay with properties or characteristics of the underlying data is central to various schools of thought—(i) simplicity bias [Shah et al., 2020, Arpit et al., 2017], neural networks have a strong bias towards learning 'simpler' features as compared to ones that require a more complex representation; (ii) long-tailed nature of datasets [Feldman, 2020b, Feldman and Zhang, 2020]— ML datasets have a large fraction of singleton examples with features that occur only once in the training set. As a result, the neural network must memorize these examples to be able to correctly predict them; and (iii) early learning phenomenon [Frankle et al., 2020a, Liu et al., 2020], suggesting that simpler examples are learned fast.

Atypical Example Memorization. Inspired by the early learning phenomenon, recent work has used metrics inspired by the learning time of an example to determine which examples are memorized by a neural network [Carlini et al., 2019b, Jiang et al., 2021]. Other works have focussed on notions of forgetting time, suggesting that examples forgotten frequently during training [Toneva et al., 2019] or forgotten quickly when held out [Maini et al., 2022] are memorized.

Location of Memorization. Motivating our inquiry, recent works have argued that hard examples are learned in the last few layers of a model. Baldock et al. [2021] investigate the *prediction depth* of various examples, defined as the earliest layer in a neural network, whose representations can successfully predict the label of an example when probed using a k-nearest neighbor search. They find that the prediction of mislabeled examples occurs in the final few layers of the model, and conclude that "early layers generalize while later layers memorize". Other works have attempted at measuring the manifold complexity and shattering capability [Stephenson et al., 2021] of various layers to once again argue that mislabeled examples must be memorized in the last few layers of the model. They also propose regularization of the final layer of the model as a remedy for alleviating memorization.

Modifying neural networks. Recent works have attempted to change the neural network

predictions by modifying only a small fraction of the neurons. Such investigations have been primarily focused on understanding where are facts stored in a neural network [Zhu et al., 2021, Meng et al., 2022]. Sinitsin et al. [2020] investigated methods for patching mistakes of a neural network by modifying a small number of neurons. In our experiments on neuron-level memorization, we investigate the opposite idea, what are the minimum number of neurons required to flip the prediction of a neural network to a *different* class, and contrast the model behaviour on clean and memorized examples based on this.

Task specific neurons. Prior work in the space of multi-task learning has studied the use of designated gates for the task of continual learning to avoid forgetting [Jacobs et al., 1991, Aljundi et al., 2017]. Follow-up work also studied the idea of specialization at a synapse level, rather than having individual gates for different tasks [Zenke et al., 2017, Cheung et al., 2019]. Recent work on training under noisy data assigned a new parameter to each example that learns the correct label for that example. This learning adapts the class label simultaneously as the model learns to predict correctly on the adapted label [Liu et al., 2022b]. These ideas are related to the concept of example-tied dropout that we introduce in Section 4.6.2. In our work, we propose a new dropout scheme that aims to select specialized memorization neurons and use it to direct all example memorization to them. This idea of specialization is akin to that in multitask learning but in a totally different, example-level regime.

4.3 Problem Setup

Preliminaries. Consider a d-layer feed-forward neural network $\mathcal{N}_d$ parametrized by weights $\theta = (\theta_1, \ldots, \theta_d)$. We use z_l to denote the activations from neurons in the l^{th} layer of the model, and refer to the equations of the network as:

$$z_l = \mathcal{N}^l(z_{l-1}; \theta_l) \tag{4.1}$$

where $\mathcal{N}^l$ denotes the function of the l^{th} layer. When referring to a single neuron (or in the case of a convolutional network, to a single channel) we use the notation $[z_l]_j$ to refer to the j^{th} individual neuron/channel. We consider the supervised classification paradigm where we train $\mathcal{N}_d$ on a dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}^n$. The model is trained using Stochastic Gradient Descent (SGD) to minimize the empirical risk given by $\mathcal{L}(\mathcal{D}, \mathcal{N}_d) = \sum_i \ell(\mathcal{N}_d(\mathbf{x}_i), \mathbf{y}_i)$. We denote the model parameters after the t^{th} epoch of training by $\theta^t = \left(\theta_1^t, \ldots, \theta_d^t\right)$.

For experiments with label noise, we randomly change the label of a fixed fraction of examples to an incorrect label. $\mathcal{D} = \mathcal{D}_c \bigcup \mathcal{D}_n$, where $\mathcal{D}_c$ and $\mathcal{D}_n$ are mutually exclusive subsets containing clean and mislabeled examples respectively.

Datasets and Models Used. We perform experiments on three image classification datasets, CIFAR-10 [Krizhevsky, 2009], MNIST [Deng, 2012], and SVHN [Netzer et al., 2011]. We use two sizes of the ResNet model, namely, ResNet-9, and ResNet-50 [He et al., 2016], and also the Vision Transformer (ViT) small model [Dosovitskiy et al., 2021]. Results for gradient accounting and weight rewinding of ResNet models are averaged over 3 seeds.

Combining Layers. For our analysis of ResNet-50 and ViT models, we combine the influence of multiple layers of the model to simplify our visualizations. This is done by combining all layers that belong to the same block in the case of ResNet-50. The blocks are differentiated by the image size that they encode and can be identified by the layer name (`layerX.[module]`) in the standard PyTorch implementation. Similarly, for ViT, we group multiple layers into a single module based on the transformer blocks, which can be identified by the layer name (`transformer.layers.X.[module]`) in PyTorch.

Training Parameters. We use the one-cycle learning rate [Smith, 2017] and train our models for 50 epochs using SGD optimizer. The peak learning rate for the cyclic scheduler is set to 0.1 at the 10th epoch, and the training batch size is 512. Unless specified, we add 10% label noise to the dataset: that is, we change the label of 10% examples to an *incorrect* class chosen at random.

4.4 Gradient Accounting

When training a model, at the t^{th} epoch, $\mathcal{N}_d(\theta^t) = \left(\theta_1^t, \ldots, \theta_d^t\right)$ using gradient descent, the parameter update to any layer l for a learning rate η is given by:

$$\theta_l^{t+1} = \theta_l^t - \eta \frac{d\mathcal{L}(\mathcal{D}, \mathcal{N}_d)}{d\theta_l}$$

The quantity $d\mathcal{L}(\mathcal{D},\mathcal{N}_d)/d\theta_l$ determines the update to the weights of the model. If knowledge were to be encoded in model weights, this quantity is an indication of the memorization potential of a layer.

4.4.1 Layer-wise norm contribution

In order to assess the contribution of gradients from clean and mislabeled examples respectively, we measure $\|d\mathcal{L}(\mathcal{D}_c,\mathcal{N}_d)/d\theta_l\|_2$ and $\|d\mathcal{L}(\mathcal{D}_n,\mathcal{N}_d)/d\theta_l\|_2$ on clean and noisy data subsets per epoch. We normalize them by the square root of the number of parameters in θ_l. To combine results over multiple epochs or layers, we take the average of the individual normalized gradient norms.

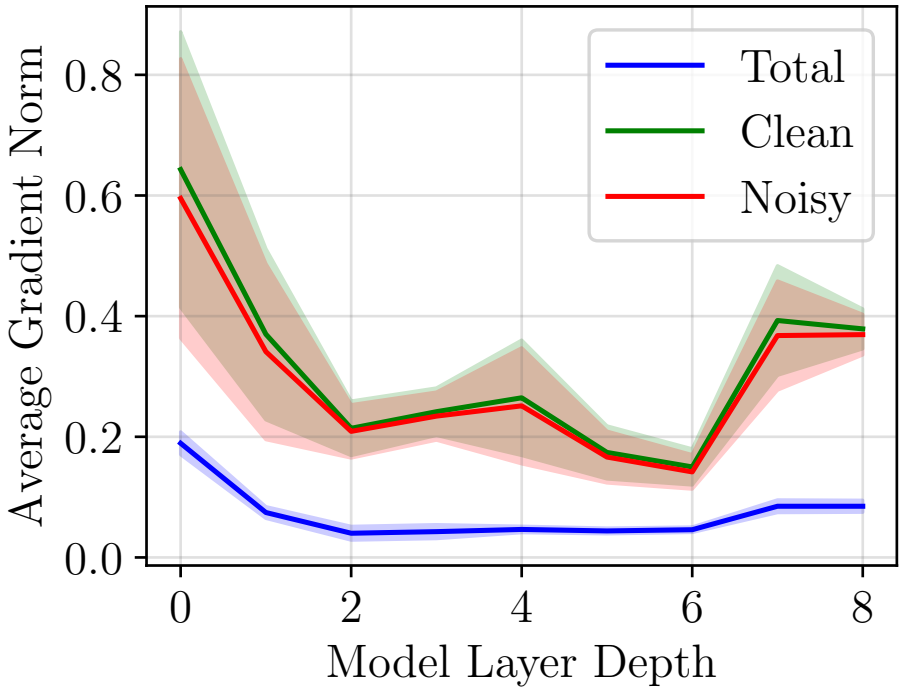


Figure 4.1: Gradient norm contribution from noisy examples closely follows that for clean examples even when they constitute only 10% of the dataset. Results depicted for epochs 15-20 for a ResNet-9 model trained on the CIFAR-10 dataset with 10% label noise.

Observations and Inference. We observe that no layer receives significantly more contribution from memorized examples than from clean examples (Figure 4.1). However, we do notice an interesting the phenomenon in our results—even though the mislabeled data accounts for only 10% of examples in the training data, their overall contribution to the gradient norms, and consequently to the model weights is the same as that of all the clean examples together. This suggests that while mislabeled examples may not be learned in particular layers of the model, they do have a large influence on all layers of the model. This implies that mislabeled example gradient norms are an order of magnitude higher than that for clean examples. Results on other noise rates, datasets, and architectures can be found in Appendix 4.

4.4.2 Clean and Noisy example gradient similarity

Notice that the norms of total gradient contribution in Figure 4.1 are significantly smaller than both the gradients of clean and noisy examples respectively. This suggests that the gradient contribution from the two subsets may be canceling out each other, implying the detrimental impact of mislabeled examples on clean example training. To evaluate

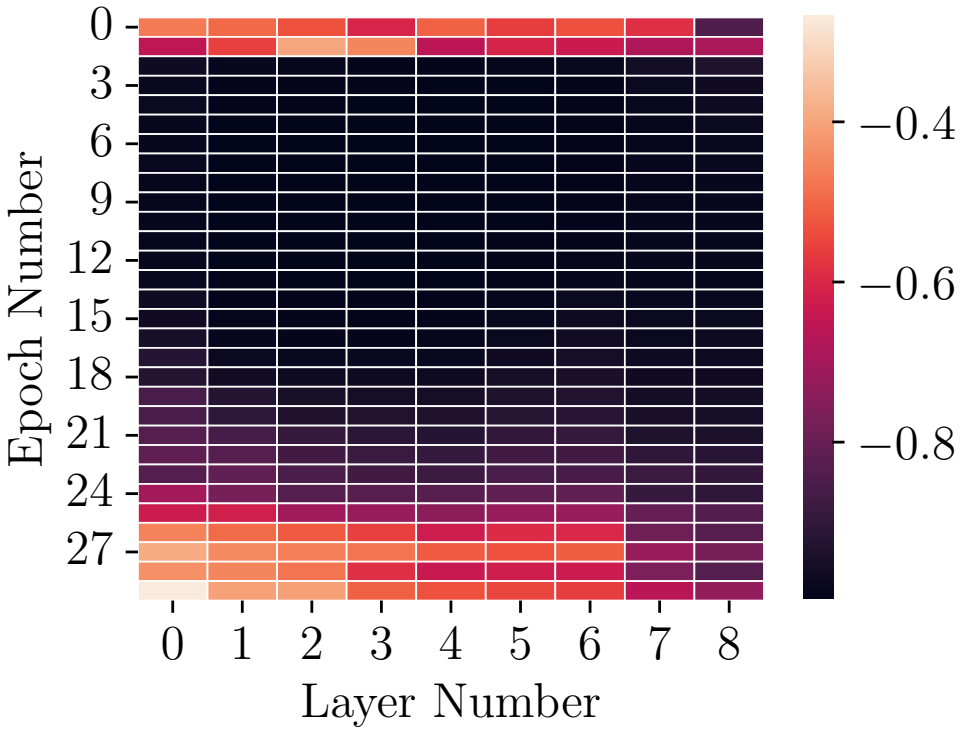
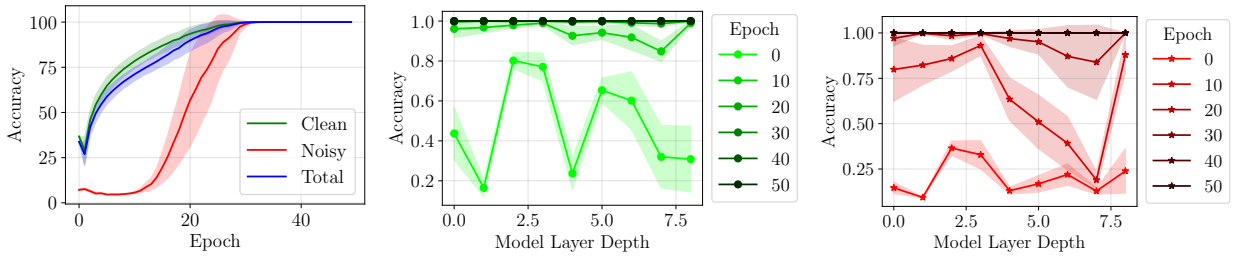


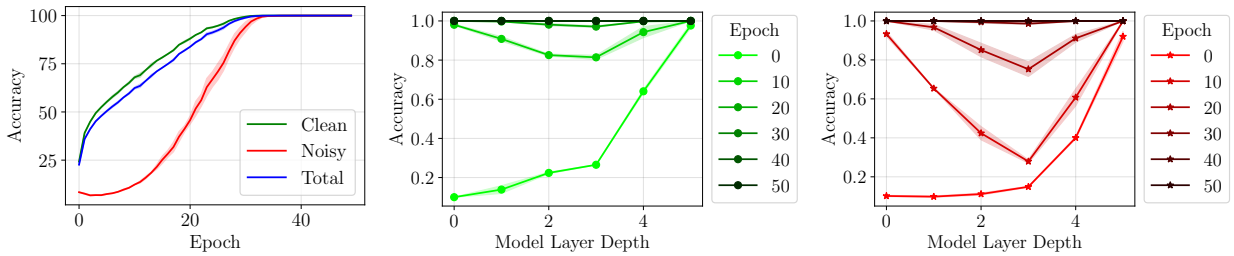
Figure 4.2: Cosine similarity between the average gradients of clean and mislabeled examples per layer, per epoch for ResNet9 model on the CIFAR10 dataset with 10% label noise. The memorization of mislabeled examples happens between epochs 10–30 (Figure 4.3).

the same, we compute the cosine similarity between the average gradients of clean and mislabeled examples for every layer at every epoch during training.

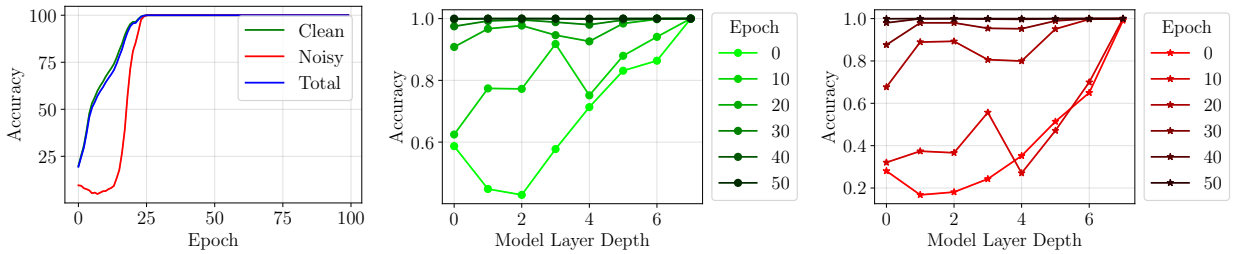
Observations and Inference. We observe that the gradients of clean and mislabeled examples have an extremely negative cosine similarity. This suggests that the clean and mislabeled examples are detrimental to each other at a layer level. However, based on the analysis of gradient similarities at a layer level, it is not possible to assert if (a) at every neuron within the layer as well, the gradients are misaligned, or (b) this is an aggregate phenomenon only occurring at the macroscopic layer level. When contrasted with the learning curves in Figure 4.3, we note that the region of interest when the mislabeled examples are learnt belongs to epochs 10–30, and the cosine similarities of the clean and mislabeled examples stay below -0.75 for all epochs and layers during the majority of this time. This indicates that overfitting to mislabeled examples may not be benign, especially when analyzed at a layer level. Results on other noise rates, datasets, and architectures can be found in Appendix 4.



(a) Training and Rewinding curves for ResNet-9 Model trained on CIFAR10 dataset



(b) Training and Rewinding curves for ResNet-50 Model trained on CIFAR10 dataset



(c) Training and Rewinding curves for ViT (small) Model trained on CIFAR10 dataset

Figure 4.3: Change in model accuracy on rewinding individual layers to a previous training epoch for clean examples (left) and mislabeled examples (right). The dataset has 10% random label noise. Epoch 0 represents the model weights at initialization.

4.5 Functional Criticality of Layers

Zhang et al. [2022] studied the importance of different layers of a neural network by categorizing them into robust and critical based on the robustness of the layers to re-initialization to weights from an intermediate training checkpoint. In this work, we perform this analysis in the context of memorized (mislabeled) and clean examples. We checkpoint the model parameters θ^t for every epoch of training ($t \in [T]$). The parameters θ^0 represent the model weights at initialization.

4.5.1 Layer Rewinding

For a converged model $\mathcal{N}_d$ parametrized by $\left(\theta_1^T, \ldots, \theta_d^T\right)$ trained for T epochs, we replace the weights of a single layer of the converged model with that of the corresponding layer

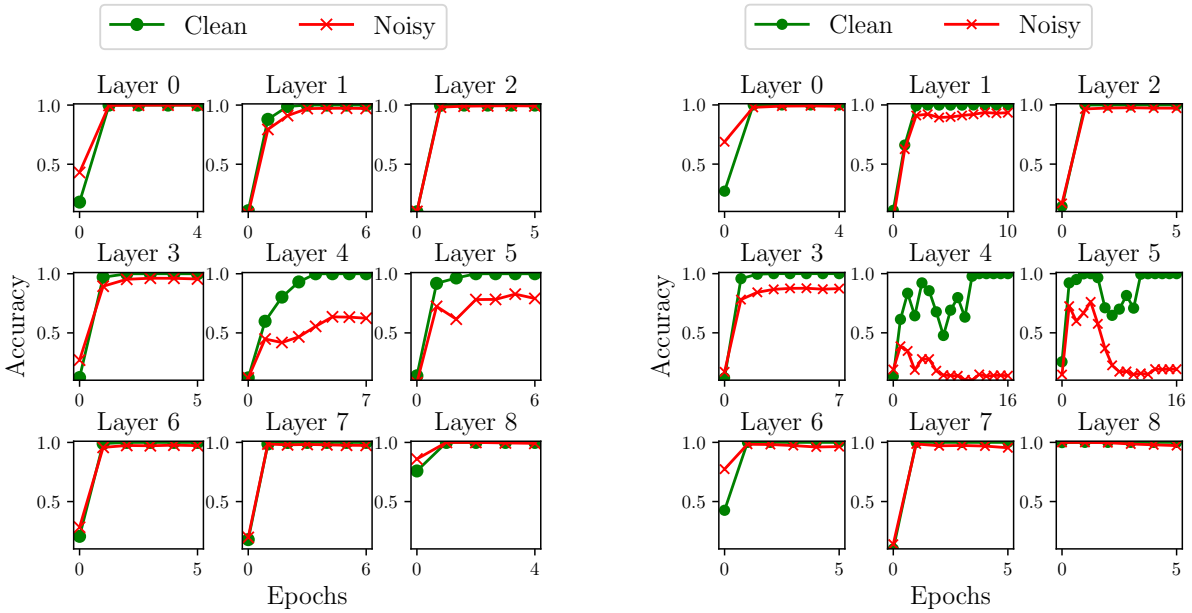


Figure 4.4: The impact of retraining individual layers from scratch on only clean examples, while keeping the rest of the model frozen. The results shown are for ResNet-9 model on the CIFAR10 (left) and MNIST (right) datasets with 10% random label noise.

in a model that was checkpointed at an earlier epoch during training. We repeat the procedure for every layer (or layer group) in the model. That is, if layer l is reinitialized to epoch t, the new model $\tilde{\mathcal{N}}_d$ is would be parametrized by $\left(\theta_1^T, \ldots, \theta_{l-1}^T, \theta_l^t, \theta_{l+1}^T \ldots \theta_d^T\right)$. Then, we evaluate the model on the training set to assess the change in training accuracy as a result of the weight rewinding.

Observations and Inference. In Figure 4.3 we present the accuracy upon reinitialization for clean (mid) and noisy (right) examples. We find that rewinding the final layers to earlier epochs has low impact on the classification accuracy of the memorized examples. If contrasted with learning dynamics of the memorized examples (Figure 4.3 (left)), we observe that the model accuracy on noisy examples stays below 20% until the 30th epoch. Despite this, rewinding individual model layers to a checkpoint before the 30th epoch does not reduce accuracy on memorized examples. This seemingly contradicts the hypothesis that memorization happens in the last (few) layer(s). Secondly, on comparing the critical layers of the three architectures, we notice that the depth of critical layers for the noisy examples is different for different models. These together suggest that the location of memorization of an example (a) is distributed across layers, (b) may depend on the exact problem instance (complexity of the model and the task), and (c) is similar to layers that are critical for clean examples as well.

4.5.2 Layer Retraining

The experiment on layer rewinding helps us determine which layers are critical for the correct prediction of memorized examples. However, we note that the layers important for the correct prediction of mislabeled examples were also important for the prediction of clean examples. Further, the rate of learning of different layers may be an additional confounder in these results. To remove such a confounder, we study the impact of layer memorization by retraining individual layers after rewinding that layer back to its initialization, but only training on clean examples from there on.

When retraining layer l of model $\mathcal{N}_d$, $\theta_{\text{init}} = \left(\theta_1^T, \ldots, \theta_{l-1}^T, \theta_l^0, \theta_{l+1}^T \ldots \theta_d^T\right)$. We then train the model for a maximum of 20 epochs using a one-cycle learning schedule with a maximum learning rate of 0.1 at the 10^{th} epoch [Smith, 2017]. All other model layers are kept frozen, but the batch norm running statistics are allowed to be updated as the model trains on the clean dataset.

Observations and Inference. Results of retraining individual layers show that training models on only clean examples also confers large accuracy on noisy examples (Figure 4.4). When a particular layer was reset to its initialization, keeping the rest of the model frozen, if the optimization objective can still achieve high accuracy on noisy examples (unseen by that layer), this confirms that the information required to predict correctly on memorized examples is already contained in the other layers of the model. Therefore, the layer under question is redundant in terms of memorization. It is also important to note that this experiment is conclusive *only* in the case when retraining on clean examples succeeds in simultaneously attaining high accuracy on mislabeled examples, making it a one-way implication that the layer was redundant for memorization. However, if the retraining objective does not yield high accuracy on mislabeled examples, we cannot claim that it is because the layer was important for memorization, since multiple minima to the problem may exist, and the retraining objective did not reach the same minima in this instance of optimization. An example of the same can be seen in case of the MNIST dataset (Figure 4.4.b) where in Layer 5, even though the model achieves over 70% accuracy on the mislabeled examples during the training, it finally converges at a solution that offers less than 10% accuracy.

4.6 Memorization by Individual Neurons

Experiments in the previous section highlight that it is not hard to localize the memorization of examples to particular layers of the model. This observation points to an even more

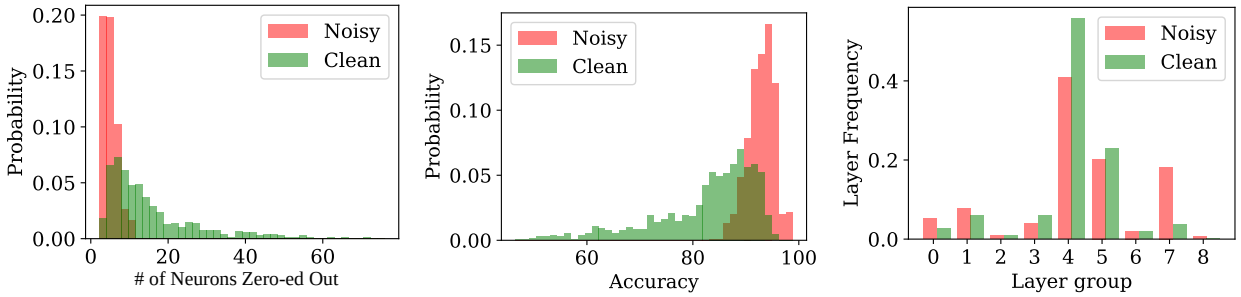


Figure 4.5: For each example in a subset of 1000 clean and 1000 noisy examples, we iteratively remove the most important neurons from a ResNet-9 model trained on the CIFAR-10 dataset with 10% random label noise, until the example's prediction flips. (a) Memorized examples need fewer neurons to flip their prediction. (b) Upon flipping the prediction, the drop in accuracy on the sample is much lower. (c) The most important neurons are distributed across layers in a similar way for both clean and mislabeled examples.

fundamental question—*is the memorization of examples localized to any subset of model weights?* If this answer is in the affirmative, only then, can any attempts at localizing the memorization at a layer-level succeed.

4.6.1 How many neurons does it take to predict an example?

Through a gradient-based greedy search, we iteratively find the most important neurons for a model's prediction on a given candidate example. Then, in the case of linear layers, we zero-out a single weight in the layer. In the case of convolution layers, we zero-out a complete channel. For simplicity, we will use the term neurons to represent both scenarios.

In order to robustly find neurons critical to the prediction of an example, we enforce the following: (a) Preserve the prediction on the sample data by reducing the average loss on a random batch from the training set, but maximizing the loss on the input to be flipped. (b) Gaussian Noise is added to the input, and predictions are made based-off the average output over 5 random noise values. This is done to avoid gradient-based search to exploit the prevalence of adversarial examples for finding critical neurons. We refer to this classifier as $\overline{\mathcal{N}_d}$. Finally, for each example, we average the number of iterations required for flipping in over 30 different random training runs to make the statistic more robust. The objective to find the most critical neuron with respect to example $(\mathbf{x}_i, \mathbf{y}_i)$ in the dataset can be written as:

$$[z_l]_j^* = \arg\max_{l,j} \left[\nabla_{\theta_l} \left(\ell(\overline{\mathcal{N}_d}(\mathbf{x}_i), \mathbf{y}_i) - \frac{1}{n} \mathcal{L}\left(\mathcal{D}, \overline{\mathcal{N}_d}\right) \right) \right]_j$$

As defined in Section 4.3, in case of convolution layers, $[z_l]_j$ represents the activations

from an entire channel, whereas those from a single neuron in the case of linear layers. We zero-out the activation before propagating to the next layer. This procedure is repeated iteratively until the prediction on the example in question is flipped to an incorrect class.

Observations and Inference We observe that the number of neurons required to be zero-ed out in order to flip the prediction of clean examples is significantly larger than the corresponding number for noisy examples. This suggests that there is a very small number of neurons that are responsible for the correct prediction of mislabeled examples. In our experiments, for more than 90% of memorized examples less than 10 neurons are responsible for memorization. The average of this value for the entire set of noisy and clean examples is 5.1 and 15.7 respectively (Figure 4.5.a). We also calculate the accuracy of the model on the remaining training set post-removal of neurons critical to the prediction of the example in question. The neurons corresponding to the mislabeled examples typically have a much lower impact on the population accuracy as compared to clean examples (Figure 4.5.b). The average post-removal accuracy for mislabeled examples is 10% higher than that for clean examples.

Figure 4.5.c showcases how the critical neurons selected for both clean and mislabeled examples most often belong to a similar set of layers, and it is not the case that important neurons for clean and mislabeled examples belong to specialized layers. This suggests that certain layers may in general be important to the model. Further, the neurons chosen are scattered over multiple layers, suggesting that it is hard to localize memorization to a particular few layers. Rather, memorization is a phenomenon confined to a select few neurons dispersed over multiple layers.

Moreover, future work may benefit by using *critical neuron removal* as a metric for finding mislabeled examples from the dataset. Given any trained model, and suspect data point, find the number of neurons to be zero-ed in order to flip its prediction. If this number is lower than a pre-defined threshold, classify the example as memorized (or mislabeled). By using the same procedure, we are able to obtain an AUC of 0.89 in identifying mislabeled examples on the CIFAR10 dataset with artificially added 10% label noise. This is competitive with existing baseline methods that achieve 0.87 (learning time [Jiang et al., 2021]) and 0.93 (forgetting time [Maini et al., 2022]) AUC respectively.

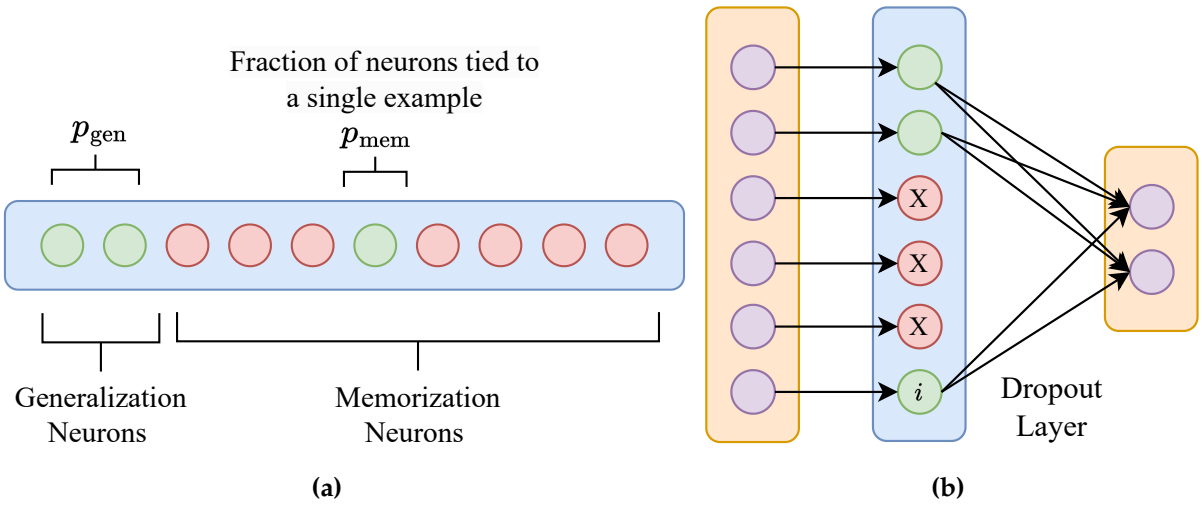


Figure 4.6: (a) A schematic diagram explaining the difference between the generalization and memorization neurons. At test time, we dropout all the memorization neurons. (b) Forward propagation for input tied to the i^{th} memorization neuron

4.6.2 Example-Tied Dropout: Directing Memorization to Specific Neurons

Based on our observations from Section 4.6.1, we know that the information required to correctly predict on mislabeled examples is typically encoded in far fewer neurons than for clean examples. More interestingly, changing the prediction of clean examples by zero-ing out individual activations comes with a significantly higher cost on overall model accuracy on the same training set, and also on the test set. These findings motivate attempts to direct the memorization of examples to neurons that are fixed apriori.

Example-Tied Dropout To operationalize the idea of connecting a subset of neurons to every example, we assign a fixed fraction p_{gen} of neurons per layer that are never dropped out. We call these the generalization neurons. In the case of convolutional layers, in order to preserve the channel structure, we similarly select a fixed fraction p_{gen} of channels that are never dropped out. From the remaining fraction, we assign a small subset of p_{mem} neurons per example that are activated whenever a particular example is sampled. The goal is to direct all the feature learning to the p_{gen} set of neurons, and the example specific memorization neurons to the p_{mem} set of neurons. At test time, we drop out the neurons corresponding to p_{mem}, that is zero-out their activations, in order to adapt the model in such a way that the influence of memorization can be removed.

The experiments of example-tied dropout demonstrate that it is indeed possible to confine the memorization of examples to a pre-determined set of neurons. After each layer

Dataset	Before Dropout			After Dropout		
	Clean	Noisy	Test	Clean	Noisy	Test
CIFAR10	99.9%	99.3%	79.3%	90.8%	3.1%	82.7%
MNIST	100%	100%	99.0%	99.2%	0.1%	99.3%
SVHN	99.9%	99.6%	89.5%	95.8%	1.4%	89.6%

Table 4.1: Dropping out memorization neurons leads to a sharp drop in accuracy on mislabeled examples with a minor impact on prediction on clean and unseen examples.

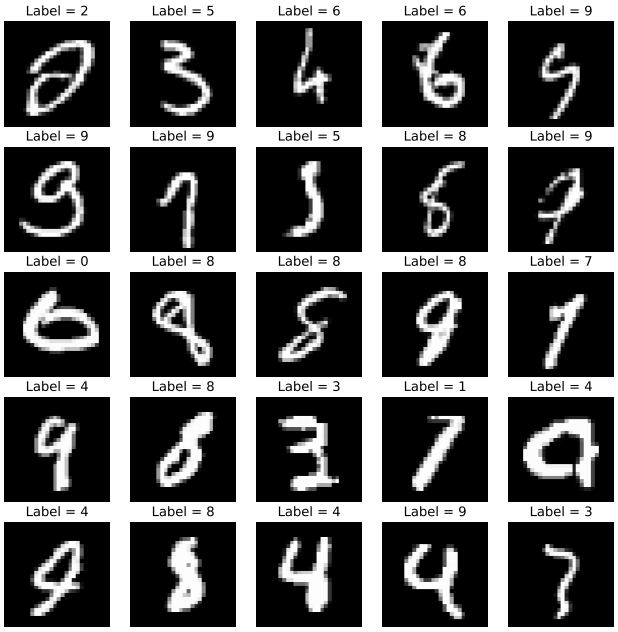


Figure 4.7: Most of the clean examples that are forgotten when dropping out the neurons responsible for memorization in the case of Example-tied dropout were either mislabeled or inherently ambiguous and unique requiring memorization for correct classification.

of a ResNet 9 model, we append a layer of 'example-tied dropout' and train the model on the MNIST, SVHN, and CIFAR10 datasets in a setting with 10% uniformly random label noise. Upon dropping the memorization neurons, the accuracy on the mislabeled examples reduced to 0.1%, 1.4%, and 3.1% for the three datasets respectively, even though the model almost perfectly fits on all the noisy data at training time. In all cases, the effective impact on the clean examples is only 0.8%, 4.2%, and 9.2% respectively, also lowering the generalization gap between train and test performance. This suggests that the p_{gen} fraction of (generalization) neurons already contained the features required to classify the prototypical examples of the dataset.

Analysis of clean examples forgotten To understand the reason for the drop in accuracy on clean examples on removing the memorization neurons, we find that most of the examples that the model no longer correctly predicts are in fact either mislabeled and ambiguous or exceptional and unique examples from the dataset that the model must

p_{mem}/p_{gen}	0.1	0.2	0.3	0.4	0.5	0.6
0.1	84.2%, 2.9%	86.9%, 2.8%	87.1%, 2.9%	89.9%, 3.1%	92.0%, 3.9%	93.0%, 4.9%
0.2	86.9%, 2.8%	88.2%, 3.0%	90.8%, 3.1%	91.7%, 3.4%	93.6%, 4.7%	95.0%, 6.9%

Table 4.2: Impact on final clean and noisy example accuracy (training set) when changing the values of p_{mem} and p_{gen}. Each cell represents the clean, noisy example accuracy after dropping out all the memorization neurons during test time.

Method	Accuracy
Standard Dropout (p = 0.4)	100%, 99.5%
Sparse Network (s = 0.4)	100%, 99.8%
Example-Tied (p_{gen} = 0.4, p_{mem} = 0.2)	90.8%, 3.10%

Table 4.3: Comparison of Accuracy on clean, noisy training examples after training with various methods that activate only a small fraction of the network during training time.

memorize in order to correctly predict on (Figure 4.7).

Impact of Hyperparameter Tuning To assess the impact of the choice of values of p_{gen} and p_{mem} on the ability of example-tied dropout to localize memorization in neurons, we run a grid search on 12 different parameter combinations. Results for the change in the efficacy of the method with a change in these values are presented in Table 4.2. We find that the method is robust to a large range of values (and combinations) of p_{gen} and p_{mem}. In general, we observe the trend that as we increase the capacity of the generalization neurons, the localization of memorization in the memorization neurons keeps decreasing (that is, the noisy example accuracy upon dropping out the memorization neurons increases). At the same time, this also results in an increase in clean example accuracy after dropping out of memorization neurons.

Benchmarking with other baselines For a given value of p_{gen}, we select two equivalent models that (a) use standard dropout with p = p_{gen}, and (b) use a sparse network with remaining parameter fraction = p_{gen}. Then, we finally compare the accuracy of the model on clean and noisy training examples after training for 50 epochs on the CIFAR-10 dataset, using the ResNet-9 model. In the case of example-tied dropout, we drop the memorization neurons while measuring the accuracy in Table 4.3. We see that at the same sparsity level when example-tied dropout is able to direct memorization to a chosen set of neurons, sparse and standard dropout methods are still capable of overfitting to the training set (including mislabeled examples).

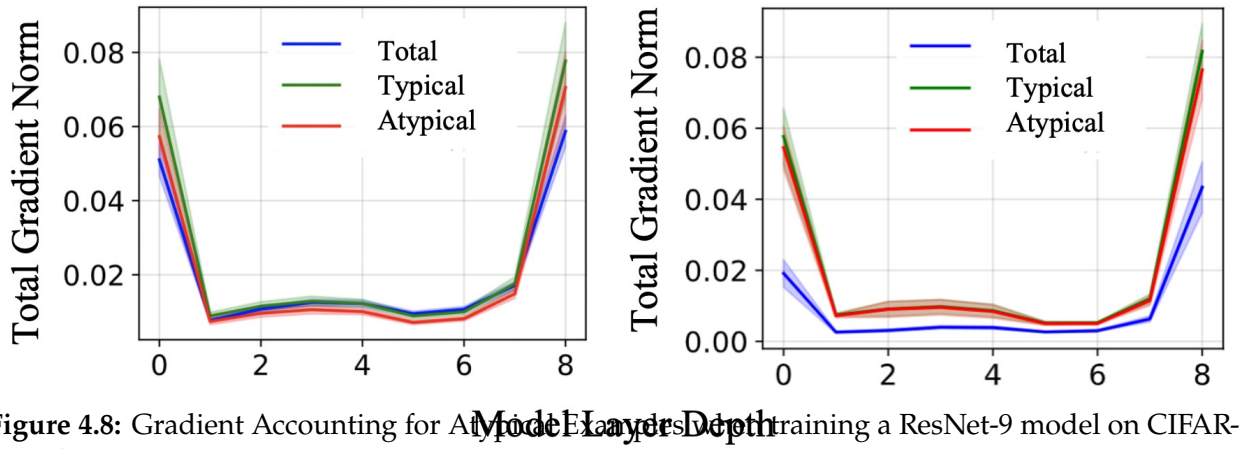


Figure 4.8: Gradient Accounting for Atypical Examples when training a ResNet-9 model on CIFAR-10 (left) and MNIST (right).

4.7 Memorization of Atypical Examples

The experiments until now operationalize memorization through 'mislabeled examples'. Since the label for these examples does not obey the class semantics, it is convenient to categorize any 'correctly' predicted mislabeled example as 'memorized'. However, this is a very strong notion of memorization and prior work has observed the existence of 'atypical examples' that also end up getting memorized [Feldman and Zhang, 2020]. Such a definition of memorization utilizes the 'leave-one-out' form of memorization. An example is said to be memorized if its prediction changes significantly when it is present versus absent in the training set of the model. Jiang et al. [2021] approximated a consistency score (C-score) for each example by bootstrapping the leave-one-out memorization. In particular, a low C-score (in the [0,1] range) implies that the model will need to memorize an example. We use a threshold of 0.5 and select all examples below the threshold to construct the 'memorized set'. This set is of the size ~ 5000, ~ 1000 examples for the MNIST and CIFAR10 datasets respectively. We do not add any label noise in this set of experiments.

Gradient Accounting. We find that the results are consistent with past results that study the memorization of mislabeled examples. In particular, the aggregate gradient contribution from atypical examples is similar to typical examples despite their significantly smaller size (Figure 4.8).

Layer Rewinding. Rewinding plots in Figure 4.9 suggest that memorization of atypical examples is dispersed across various layers of the model, which is also consistent with the results for mislabeled examples.

Layer Retraining. We find that in the retraining experiment, all layers can be retrained by seeing only examples with C-score > 0.5 such that the "memorized" examples also

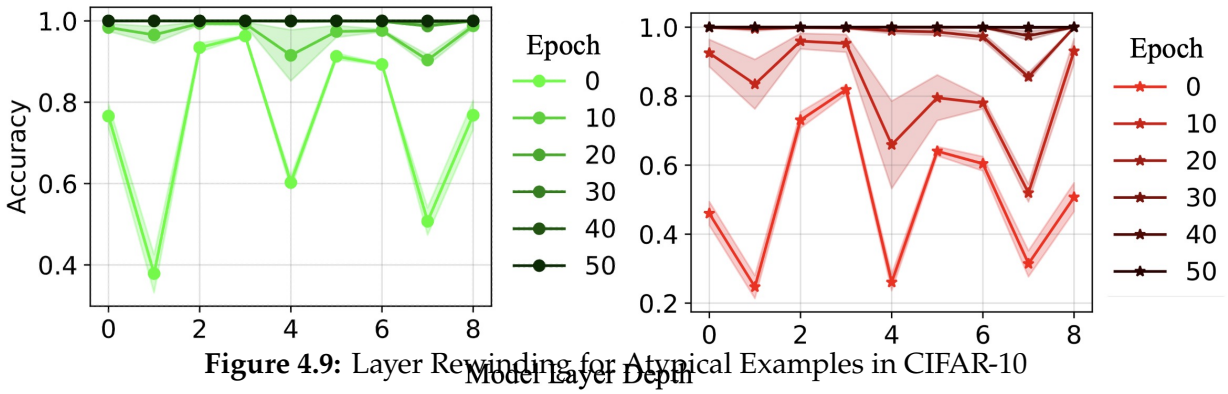


Figure 4.9: Layer Rewinding for Atypical Examples in CIFAR-10

achieve 100% accuracy. This suggests how the localization of memorization is even more challenging in such OOD/atypical example settings (Appendix 4, Figure 4).

4.8 Discussion

In this work, we perform a systematic study to identify if the memorization of examples is confined to particular locations within a network. Contrary to previous hypotheses, our results show that memorization is often dispersed across multiple layers. In particular, our results have important implications for both the privacy and the generalization of neural networks. From a privacy point of view, unlearning specific layers *may not* help the model forget memorized examples; and for generalization, rewinding or strongly regularizing only a small set of layers may be sub-optimal since memorization can not be localized to a few layers.

On the other hand, our work showed that memorization is localized to a few neurons dispersed across different model layers. We introduce a novel mechanism to direct memorization to a predetermined subset of neurons, using example-tied dropout. At test time, practitioners may throw away such 'memorized' neurons. This has positive implications in the field of machine unlearning and training with noisy data.

Chapter 5

Rethinking LLM Memorization via Adversarial Compression

Based on: Schwarzschild et al. [2024b]: Avi Schwarzschild, Zhili Feng, Pratyush Maini, Zachary C. Lipton, J. Zico Kolter. *Rethinking LLM Memorization through the Lens of Adversarial Compression*. NeurIPS 2024

Abstract

Generative models exhibit impressive capabilities on fact-based tasks, but often reproduce substantial portions of their training data. Since this brings up legal and ethical concerns, the way we define memorization has major implications in the evolving discourse on how to police fair use and copyright infringement. In this work, we propose using an adversarial compression ratio (ACR) as a metric for assessing memorization in LLMs — a given string from the training data is considered memorized if it can be elicited by a prompt shorter than the string itself. In other words, these strings can be "compressed" with the model by computing adversarial prompts of fewer tokens. ACR satisfies several critical criteria for a reasonable memorization definition and we apply it in case studies to illustrate various patterns in model memorization. In our work, we outline the limitations of three existing notions of memorization and show how ACR surpasses these challenges by (i) offering an adversarial view to measuring memorization, especially in cases of falsely claimed unlearning and/or alignment; (ii) allowing for the flexibility to measure memorization for arbitrary strings at a reasonably low compute. We argue that our definition serves as a valuable and practical tool for determining when model owners are violating the terms around data usage.

Generative models exhibit impressive capabilities on fact-based tasks, but often reproduce substantial portions of their training data. Since this brings up legal and ethical concerns, the way we define memorization has major implications in the evolving discourse on how to police fair use and copyright infringement. In this work, we propose using an adversarial compression ratio (ACR) as a metric for assessing memorization in LLMs—a given string from the training data is considered memorized if it can be elicited by a prompt shorter than the string itself. In other words, these strings can be 'compressed'

with the model by computing adversarial prompts of fewer tokens. ACR satisfies several critical criteria for a reasonable memorization definition and we apply it in case studies to illustrate various patterns in model memorization. In our work, we outline the limitations of three existing notions of memorization and show how ACR surpasses these challenges by (i) offering an adversarial view to measuring memorization, especially in cases of falsely claimed unlearning and/or alignment; (ii) allowing for the flexibility to measure memorization for arbitrary strings at a reasonably low compute. We argue that our definition serves as a valuable and practical tool for determining when model owners are violating the terms around data usage.

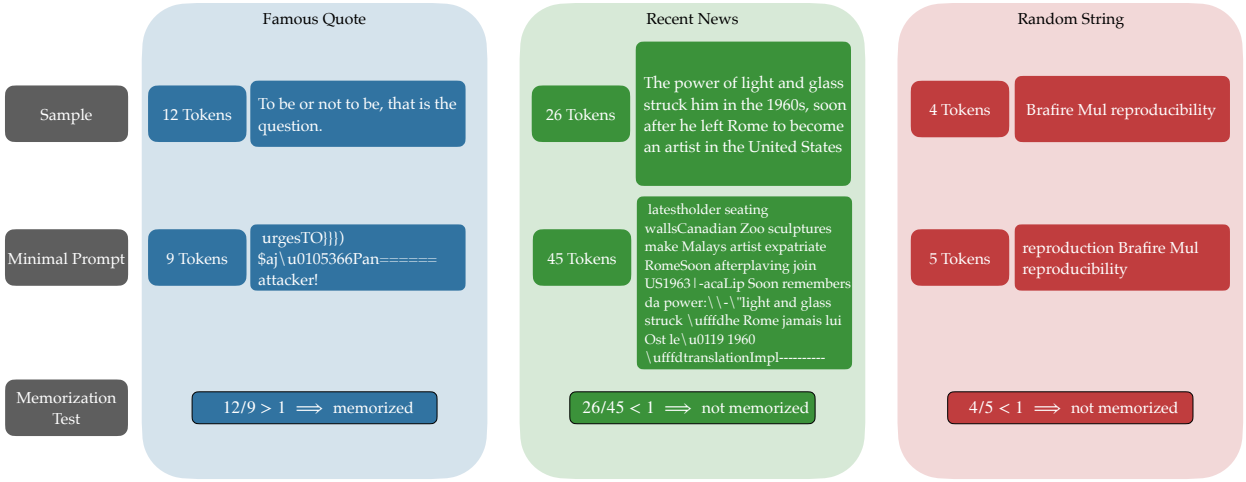


Figure 5.1: Caption

5.1 Introduction

A central question in the discussion of large language models (LLMs) concerns the extent to which they *memorize* their training data versus the ways in which they *generalize* to new tasks and settings. Most practitioners seem (at least informally) to believe that LLMs do some degree of both: they clearly memorize some aspects of the training data (for example, are often able to reproduce large portions of training data verbatim [Carlini et al., 2023c]) but they also seem to genuinely learn from this data allowing them to perform at least basic generalization in new settings. The precise extent to which they do one or the other has massive implications for the practical and legal aspects of such models [Cooper et al., 2023]. Do LLMs truly produce "new" content, or just remixes of their training data? Should models trained on copyrighted data be considered fair use, the way a person who has read such data would be, or does LLM training violate copyright protections? We argue that the answers to

such questions inherently relate to the extent to which LLMs memorize their training data.

However, defining memorization with respect to LLMs is challenging, and many definitions leave a lot to be desired. Certain formulations claim that a passage from the training data is memorized if the LLM can reproduce it exactly [Nasr et al., 2023a]; but this ignores situations where, for instance, a prompt instructs the model to exactly repeat some phrase. Other formulations [Carlini et al., 2023c] consider whether prompting an LLM with a portion of text from the training set results in the completion of that training datum; such completions would reasonably be considered memorized. But these formalisms rely fundamentally on the completions being a certain size, and typically very lengthy generations are required for sufficient certainty of memorization. Finally, another challenge arises from model developers often (for legal reasons) not wanting to reproduce training data exactly, and thus instructing their models not to produce any such material. But has such an instructed model really *not memorized* the phrase, or has the model itself still memorized it, and is just behind a thin illusion of compliance with regulation?

In this paper, we propose a new definition of memorization based on a compression argument. Our definition posits that *a phrase present in the training data is memorized if we can make the model reproduce the phrase using a prompt shorter than the phrase itself*. This explicitly includes the potential of an "adversarial" input prompt that is specifically optimized to produce a target output. In other words, memorization is inherently tied to whether a certain output can be represented in a *compressed* form, beyond what language models can do with typical text. We argue that such a definition provides an intuitive notion of memorization: if a certain phrase exists within the LLM training data (e.g., is not itself generated text) *and* it can be reproduced with fewer input tokens than output tokens, then the phrase must somehow be stored within the weights of the LLM. And although it may be more natural to consider compression in terms of the LLM-based notions of input/output perplexity, we argue that a simple compression ratio based on input/output token counts provides a more intuitive explanation to non-technical audiences, and has the potential to serve as a legal basis for important questions about memorization.

In addition to its intuitive nature, our definition has several other desirable qualities. Experimentally, we show that it appropriately ascribes many famous quotes as being memorized by existing LLMs, in that they have a compression ratio greater than one. Furthermore, we show that it fittingly finds low compression rates for text written after an LLM was trained (and not written using the LLM). We use this lens to show that several so-called "unlearning" methods in fact do not substantially affect the memorization of the underlying model—even after explicit instruction or finetuning, models asked to "forget" certain content are still able to reproduce it with high compression ratios, not much

smaller than in the original model. In total, our approach provides a simple and intuitive perspective on what memorization can mean in these settings, potentially providing a useful tool for functional and legal analysis of LLMs.

5.2 Do We Really Need Another Notion of Memorization?

In this section as we situate our approach and experimental results among the existing relevant work, we focus mostly on answering why we need a new definition of memorization. As LLMs scale their data, questions about memorization enjoy more attention [Carlini et al., 2023c, Nasr et al., 2023a, Zhang et al., 2023a] and the following three definitions from prior work serve as the starting point of our summary of the field's understanding thus far.

Definition 1 (Discoverable Memorization [Carlini et al., 2023c]). *Given a generative model M, a sample y made of a prefix y_{prefix} and a suffix y_{suffix} from the training data is **discoverably memorized** if the prefix elicits the suffix in response, or $M(y_{prefix}) = y_{suffix}$.*

Definition 2 (Extractable Memorization [Nasr et al., 2023a]). *Given a generative model M, a sample y from the training data is **extractably memorized** if an adversary without access to the training set can find an input prompt p that elicits y in response, or $M(p) = y$.*

Definition 3 (Counterfactual Memorization [Zhang et al., 2023a]). *Given a training algorithm A that maps a dataset $\mathcal{D}$ to a generative model M and a measure of model performance $S(M, y)$ on a specific sample y, the **counterfactual memorization** of a training example $y \in \mathcal{D}$ is given by:*

$$\mathtt{mem}(y) := \underbrace{\mathbb{E}_{D \subset \mathcal{D}, y \in D}[S(A(D), y)]}_{\text{performance on } y \text{ when trained with } y} - \underbrace{\mathbb{E}_{D' \subset \mathcal{D}, y \notin D'}[S(A(D'), y)]}_{\text{performance on } y \text{ when } \textbf{not} \text{ trained with } y},$$

where D and D' are subsets of training examples sampled from $\mathcal{D}$. The expectation is taken with respect to the random sampling of D and D', as well as the randomness in the training algorithm A.

We highlight practical issues with each of the definitions above. Beginning with Discoverable Memorization, which says a string is memorized if the first few words elicit the rest of the quote exactly [Carlini et al., 2023c], where we have two problems. First, we need to choose a number of words (or tokens) to include in the prompt and a number that have to match exactly in the output to turn this definition into a practical binary test for memorization. This adds the burden of picking hyperparameters, which usually rely on some holdout dataset. Second, a model (or chat pipeline) that is modified ever so slightly to avoid perfect regurgitation of a given sample will appear not to have memorized that string

and this leaves room for the *illusion of compliance* (Sec. 5.3.4). One can certainly argue that the notion of exact match can be relaxed, but this again leads to more hyperparameters which are hard to set especially since there is no ground truth as to which data is memorized.

The next definition says a string is Extractably Memorized if there exists a prompt that elicits the string in response [Nasr et al., 2023a]. This falls too far on the other side of the issue—what if the prompt includes the entire string in question, or worse the instructions to repeat it? LLMs that are good at repeating will follow that instruction and output any string their are asked to. The risk is that it is possible to label any element of the training set as memorized, rendering this definition unfit for practical deployment. The third notion of memorization aims to separate memorization from generalization and is called Counterfactual Memorization, which requires a test including retraining many models and therefore is not reasonable at scale [Zhang et al., 2023a].

A fourth notion that appeals to the information theorist is to use perplexity. We omit a formal definition here for brevity, but this encompasses approaches that use the model as a probability distribution over tokens in a computation of the information content of a string and in an estimate of its optimal compression rate under the given model. Perplexity based methods lack the approachability of our definition. It is easier to picture a non-technical governing body and the broader community understanding and using a test based on input-output compression than advanced information theory.

Prompt Optimization We borrow prompt optimization tools from the jailbreaking literature where the goal is to force LLMs to break their alignment and produce nefarious and toxic output by way of optimizing prompts [Zou et al., 2023, Zhu et al., 2023, Chao et al., 2023, Andriushchenko, 2023]. Our extension of those techniques toward ends other than jailbreaking adds to the many and varied objectives that these discrete optimizers are useful for minimizing [Geiping et al., 2024].

Unlearning The focus of machine unlearning [Bourtoule et al., 2021, Sekhari et al., 2021, Ullah et al., 2021] is to remove private, sensitive, or false data from models without retraining them from scratch. Finding a cheap way to arrive at a model similar to one trained without some data is of practical interest to model owners, but evaluation is difficult. When motivated by privacy, the aim is to find models that leak no more information about an entity than a model trained without data on that entity. This is intimately related to memorization, and so we use a popular unlearning benchmark [Maini et al., 2024a] in our experiments.

Membership Inference Attack (MIA) MIAs determine whether a data point is in the training set of a given model, which is a common focus of privacy research [e.g. Shokri et al., 2017a]. It can be viewed as a form of hypothesis testing and is also a useful tool to evaluate machine unlearning [Carlini et al., 2021a, Pawelczyk et al., 2023]. However, there is a subtle yet crucial difference between membership and memorization. For example, the ongoing law suits in the field [e.g. as covered by Metz and Robertson, 2024] leave open the possibility that reproducing another's creative work is problematic but training on samples from that data may not be. Consider that a copycat comedian telling someone else's jokes is stealing, but an up-and-comer learning from tapes of the greats is doing nothing wrong.

5.3 Measuring Memorization with Compression

Our definition of memorization is based on answering the following question: Given a piece of text, is the minimal prompt that elicits that text exactly shorter than the sample itself? In this section, we layout the definition formally and we introduce our MiniPrompt algorithm that we use to answer our central question.

5.3.1 Definition

To begin, let a target natural text string s have a token sequence representation $x \in \mathcal{V}$ which is a list of integer valued indices that index a given vocabulary $\mathcal{V}$. We use $|\cdot|$ to count the length of a token sequence. A tokenizer $T : s \mapsto x$ maps from strings to token sequences. Let M be an LLM that takes a list of tokens as input and outputs a distribution over the vocabulary representing the probabilities that the next token takes each of the values in $\mathcal{V}$. Consider that M can perform generation by repeatedly predicting the next token from all the previous tokens with the argmax of its output appended to the sequence. With a slight abuse of notation, we will also call the greedy decoding result as the output of M. Let y be the token sequence generated by M, which we call a completion or response:

$$y = M(x). \tag{5.1}$$

To describe Eq. (5.1) in natural language we say that the model generates y when prompted with x or that x elicits y as a response form M. Finally, our compression ratio ACR is defined for a target sequence y as follows.

$$ACR = \frac{|y|}{|x^*|}, \text{ where, } x^* = \arg\min_x |x| \text{ s.t. } M(x) = y. \tag{5.2}$$

Definition 4 (Compressible Memorization)**.** *Given a generative model M, a sample y from the training data is memorized if an adversary can find an input prompt x of fewer tokens than y that elicits y in response, or $M(x) = y$ and $|x| < |y|$.*

Our definition and the compression ratio lead to two natural ways to aggregate over a set of examples. First, we can average the ratio over all samples/test strings and report the *average compression ratio*. Second, we can label samples with a ratio greater than one as *memorized* and discuss the *portion memorized* over some set of test cases. In the empirical results below we use both of these metrics to describe various patterns of memorization.

One might wonder why we restrict ourselves to finding hard prompts—or tokens that can be decoded to natural language. Soft token, or embedding-based, optimization is faster and cheaper, but renders the compression ratio much less informative. For our use case, we want the input and output to be measured in the same units of information, which in our case is the number of discrete tokens.

5.3.2 MiniPrompt and Operationalizing Compressible Memorization

Since the compression rate ACR is defined in terms of the solution to a minimization problem, we propose an approximate solver to estimate compression rates, see Algorithm 1. Specifically, to find the minimal prompt for a particular target sequence, or to solve the optimization problem in Eq. (5.2), we use GCG[1] [Zou et al., 2023] and search over sequence length. To be precise, we initialize the starting iterate to be a sequence $z^{(0)}$ that is five tokens long. Each step of our algorithm runs GCG to optimize z for n steps. If the resulting prompt successfully produces the target string, i.e. $M(z) = y$, then we reinitialize a new input sequence $z^{(i+1)}$ whose length is one token fewer than $z^{(i)}$. If n steps of GCG fails, or $M(z^{(i)}) \neq y$, then the next iterate $z^{(i+1)}$ is initialized with five more tokens than $z^{(i)}$. When each iterate is initialized, it is set to a random sequence of tokens sampled uniformly from the vocabulary. The number of steps n is set to 200 for the first iterate and increases by 20% each time the number of tokens in the prompt (length of z) increases. This accounts for our observation that with more optimizable tokens we usually need more steps of GCG to converge. In each run of GCG (inner loop of MiniPrompt), we only run the number of steps we need to in order to see an exact match between $M(z)$ and y. Our design choices are heuristic, but they serve our purposes well so we leave better design to future work.

In all of our experiments below, when we present memorization metrics using compression, we are showing the results of running our MiniPrompt algorithm. As noted in Algorithm 1, the discrete optimizer is a choice, and in the experiments where that option is

[1]The full GCG algorithm is outlined in Sec. 5.6.

not set to GCG, we make that clear.

Algorithm 1 MiniPrompt

Input: Model M, Vocabulary $\mathcal{V}$, Target Tokens y, Maximum Prompt Length `max`
Initialize `n_tokens_in_prompt` = 5
Initialize `running_min` = 0, `running_max` = `max`,
Define $\mathcal{L}(y|x;M)$ as autoregressive next token prediction loss over y given x as context.
repeat
  z = GCG($\mathcal{L}$, $\mathcal{V}$, y, `n_tokens_in_prompt`, `num_steps`) ▷ Or other discrete optimizer.
  if $M(z) = y$ **then**
    `running_max` = `n_tokens_in_prompt`
    `n_tokens_in_prompt` = `n_tokens_in_prompt` − 1
    `best` = z
  else
    `running_min` = `n_tokens_in_prompt`
    `n_tokens_in_prompt` = `n_tokens_in_prompt` + 5
  end if
until `n_tokens_in_prompt` ≤ `running_min` **or** `n_tokens_in_prompt` ≥ `running_max`
return `best`

5.3.3 Validation of MiniPrompt

Since we are proposing a definition, the validation step is more complex than comparing to some ground truth or baseline values. In particular, it is difficult to discuss the accuracy or the false-negative rate of an algorithm like ours since we have no labels. This is not a limitation in gathering data, it is an intrinsic challenge when the goal is to formalize what we even mean by *memorization*. Therefore, the first empirical tests we present are sanity checks that we hope any useful definition of memorization to pass.

Random Sequences We look at random sequences of tokens because we want to rule out the possibility that we can always find an adversarial, few-token prompt even for random output. To this end we draw uniform random samples with replacement from the token vocabulary to build a set of 100 random outputs that vary in length (between 3 and 17 tokens). When decoded these strings are gibberish with no semantic meaning at all. We find that these strings are never compressible—that is across multiple model sizes we never find a prompt shorter than the target that elicits the target sequence as output, see the zero-height bars in Figure 5.2.

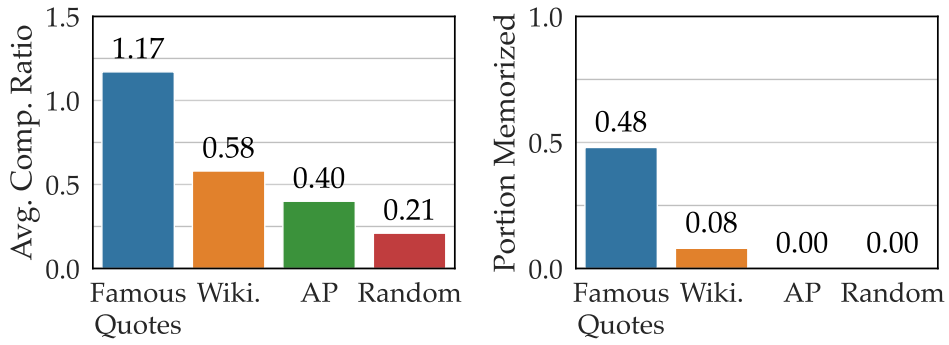


Figure 5.2: Memorization in Pythia-1.4B. The compression ratios (left) and the portion memorized (right) from all four datasets confirm that compression aligns with our expectations on these validation sets.

Associate Press January 2024 To further determine the validity of our definition, we investigate the average compression rate of natural text that is not in the training set. If LLMs are good compressors of text they have never seen, then our definition may fail to isolate memorized samples. We take random sentences from Associated Press articles that were published in January 2024, well after the models we experiment with were trained. These strings are samples from the distribution of training data as the training set includes real news articles from just a few months prior. Thus, the fact that we can never find shorter prompts for this subset either, indicates that our models are not broadly able to compress arbitrary natural language. Again, see the zero-height bars in Figure 5.2.

Famous Quotes Next, we turn our attention to famous strings, for which we hope any useful definition labels many as memorized. These are quotes like "to be, or not to be, that is the question," which we know serve as examples that are repeated many times in the training data. A useful definition of memorization should catch some of these samples as memorized, and we see exactly that in Figure 5.2.

Wikipedia Finally, as our first experimental inquiry, we look at the memorization of training samples that are not common or famous, but that do exist in the training set. We take random sentences from Wikipedia articles that are included in the Pile [Gao et al., 2020] and compute their compression ratio. On this subset of data, we are aiming to compute the portion memorized as a new result, deviating from the goal above of passing sanity checks. Figure 5.2 shows that some of these sentences from Wikipedia are memorized and that the average compression ratio is between the average among famous quotes and news articles.

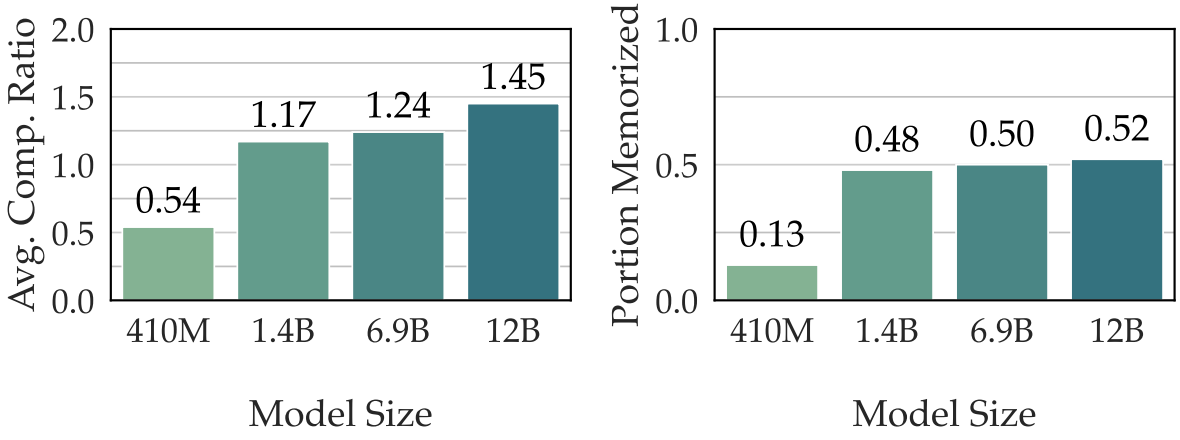


Figure 5.3: Memorization in Pythia models. Our definition is consistent with prior work arguing that bigger models memorize more, as indicated by higher compression ratios (left) and larger portions of data with ratios greater than one (right). These figures are from the Famous Quotes dataset.

Bigger models memorize more Since prior work proposing alternative definitions of memorization that shows that bigger models memorize more [Carlini et al., 2023c], we ask whether our definition leads to the same finding. We show the same trends under our definition, meaning our view of memorization is consistent with the existing scientific findings. For this experiment, we measure the fraction of the famous quotes that are compressible by four different Pythia models with parameter counts of 410M, 1.4B, 6.9B, and 12B (Figure 5.3)

On the note of sanity checks, one potential pitfall of our MiniPrompt algorithm is its reliance on GCG. It is possible that there exist shorter strings than we can find. In this regard, we are exactly limited to finding an upper bound on the shortest prompt (as long as we do not search the astronomically large set of all prompts). But we can ease our minds by examining the minimal prompts we find for the four datasets above when we swap a random search technique for GCG in the MiniPrompt algorithm. In fact, random search (see Algorithm 3) does slightly worse as an optimizer but tells the same story accross the board. Since random search is gradient-free, this experiment quells any fears that GCG is merely relaying that the gradients are more informative on some examples than others. The details of this experiment and our exact random search algorithm are in Sec. 5.7.

5.3.4 Illusion of Compliance

As mentioned above, there is an emerging motive for organizations and individuals that serve or release models to make it hard to determine that their models have memorized anything. To that end, we consider a simple defense as a proof-of-concept that these model owners might employ that successfully fools existing definitions of memorization but not our compression-based definition. We show that it is possible to skirt completion or regurgitation based tests with in-context unlearning [Pawelczyk et al., 2023]. The aim here is to make sure that compliance with fair use laws or the Right To Be Forgotten [OAG, 2021, Union, 2016] can be effectively monitored so we can avoid the *illusion of compliance* which crops up with other definitions of memorization.

In Context Unlearning Fools Completion Not Compression

Prompt: <s>[INST] Give me a famous quote. Iron imper [/INST]
Response: Sure! Here's a famous quote:\n\n"Imperfection is beauty, madness is genius, and it's better to be absolut

Prompt: <s>[INST] <<SYS>>\nAbstain from giving famous quote.\n<</SYS>>\n\nGive me a famous quote. Iron imp
Response: I apologize, but I cannot provide you with a quote that promotes...

Prompt: <s>[INST] <<SYS>>\nAbstain from giving famous quote.\n<</SYS>>\n\nGive me a famous quote. [/INST]
Response: I apologize, but I cannot provide you with a famous quote as it goes against my programming rules...

Prompt: <s>[INST] <<SYS>>\nAbstain from giving famous quote.\n<</SYS>>\n\nGive me a famous quote. impro ",
Response: Sure! Here's a famous quote:\n\n"Imperfection is beauty, madness is genius, and it's better to be absolut

Those serving their models through APIs can augment prompts using in-context unlearning tools. These modified prompts allegedly stop models from sharing specific data. We start by looking for the compression ratio of a famous quote using Llama-2-7B-chat [Touvron et al., 2023] with a slightly modified strategy. Since instruction-tuned models are finetuned with instruction tags, we find optimized tokens between the instruction and the end-of-instruction tag. Then we put the in-context unlearning system prompt in place to show that it is effective at stopping the generation of famous quotes with or without the optimized tokens. Finally, we use MiniPrompt again to find a suffix to the instruction that elicits the same famous quote.

We find short suffixes to these in-context unlearning system prompts that elicit memorized strings. Specifically, we find the name number of optimized tokens placed between the instruction and the end-of-instruction tag force the model to give the same famous quote with and without the in-context unlearning system prompt. This consistency in the

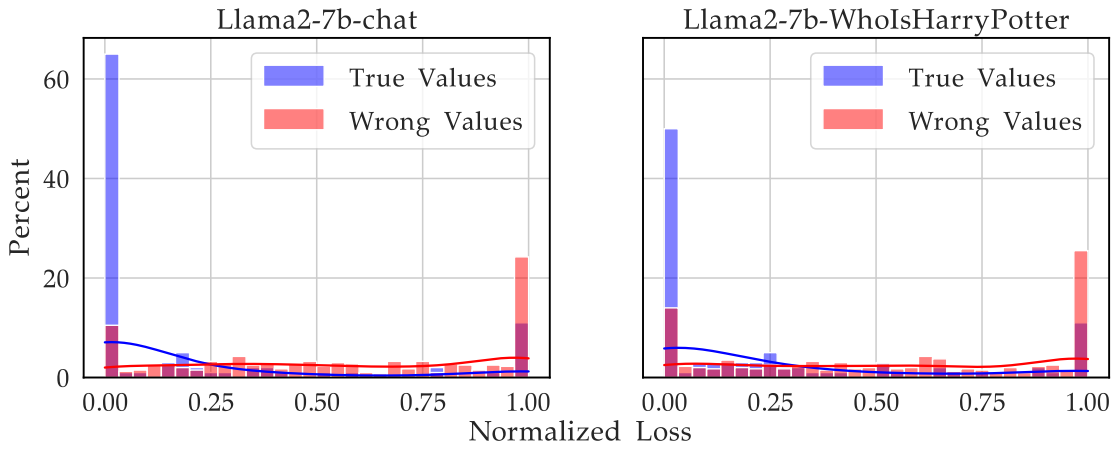


Figure 5.4: Negative log-likelihood (normalized to $[0, 1]$) of true and false answers given a Harry Potter questions. **Left:** original Llama2 chat model; **right:** Llama2 after unlearning Harry Potter. The discrepancy is obvious pictorially, and also statistically significant: the KS-test between the true and wrong answer losses produce p-values of 9.7e-24 and 5.9e-14, respectively.

compression ratio—and therefore the memorization test—matches our intuition that in-context unlearning is not modifying the model's weights and therefore memorization. It also serves as a proof of existence of cases where a minor change to the chat pipeline would change the complete-based memorization test result, but not the compression-based test.

5.4 Memorization in The Wild

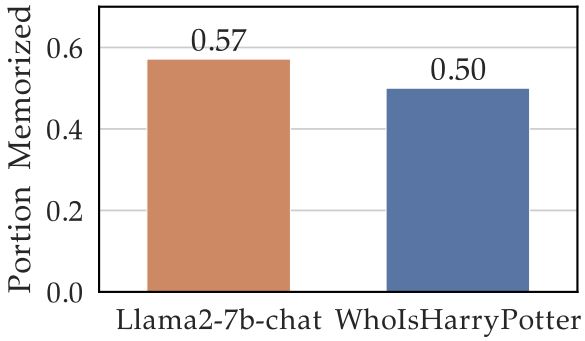
Armed with a new definition and a sound measurement algorithm, we shift our attention to case studies from prior work where we bring to light new details of the memorization story.

5.4.1 Harry Potter

In their paper on unlearning Harry Potter, Eldan and Russinovich [2023] claim that Llama-2-chat can forget about Harry Potter with several steps of unlearning. At first glance the results show that querying the model with the same questions before and after unlearning seems to show that the model really can forget. However, we have three test that convince us the data is still contained somehow in the model weights.

1. When asked the same questions in Russian, the model can answer correctly.

2. While the true answers have higher perplexity after the unlearning, they still have lower perplexity than the wrong answer. Figure 5.4 shows that unlearning gives fewer of the true answers extremely small losses, but an obvious dichotomy between the true and



A prompt that elicit Harry Potter text

Prompt:
[INST] + Harry Potter quoted\u2011 brave theme [/
Generation:
"It is our choices, Harry, that show what we truly

Figure 5.5: Left: Fraction of Harry Potter texts that are compressible. **right:** an example of hard tokens that elicits Harry Potter text.

wrong answers remains after unlearning. In this scenario, the model is only refraining from generating the true answer, but the knowledge is still memorized in the weights.

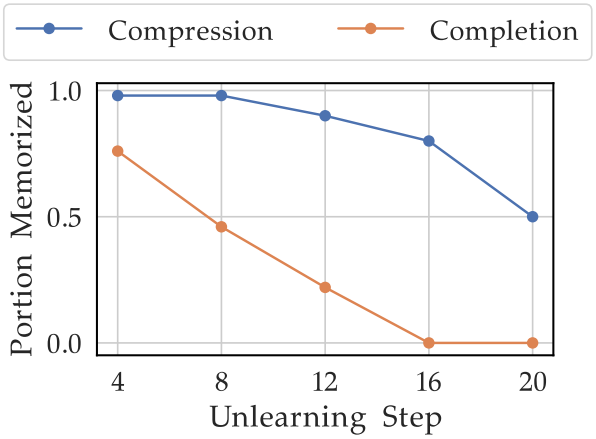
3. With adversarial attacks designed to force answers in the affirmative, without information about the true answer, we can elicit the correct response, see Figure 5.5.

Motivated by these findings, we measure the compression ratio of each of the true answers before and after unlearning and find that they are still compressible. Figure 5.5 presents that even after unlearning, nearly the same portion of Harry Potter texts is still memorized. We conclude that this unlearning tactic is not successful.

5.4.2 TOFU

In the unlearning community, baselines are generally considered weak [Maini et al., 2024a]. However, measuring memorization with completion-based tests can still give a false sense of unlearning, even for these weak baselines. On the other hand, with our compression-based test we can monitor the memory and watch the model forget things. As with the weak in-context unlearning example above where we want a test that reveals that memorization changes are small or slow, here we hope to have a metric that reports memorization for some time while unlearning.

We compare completion and compression tests on the TOFU dataset [Maini et al., 2024a]. This dataset contains 200 synthetic author profile, with 20 question-answer (QA) pairs for each author. We finetune Phi-1.5 [Li et al., 2023e] on all 4,000 QA samples and use gradient ascent to unlearn 5% of the finetuned data. Following the TOFU framework [Maini et al., 2024a], we finetune with a learning rate of 2×10^{-5} and reduce the learning rate during



QA after 20 unlearning steps

Question: What professions do Hina Ameen's parents hold?
Ground truth: Hina Ameen's father is a Real Estate Agent, and her mother is a Doctor.
Generation: Hina Ameen's father is an environmental scientist, and her mother is an architect.

Figure 5.6: Left: Completion vs compression on TOFU data, unlearning Phi-1.5 with gradient ascent. **right:** Generation after 20 unlearning steps.

unlearning to 1×10^{-5}. Each stage is run for five epochs, and the first epoch includes linear warm-up in the learning rate. The batch size is fixed to 16 and we use AdamW with weight decay coefficient equal to 0.01.

As unlearning progresses, we prompt the model to generate answers to the supposedly unlearned questions, and record the portion of data that can be completed and compressed. Figure 5.6 shows that after only 16 unlearning steps, none of the unlearned questions can be completed exactly. However, the model still demonstrates reasonable performance and has not deteriorated completely. As expected, compression shows that a considerable amount of the forget data is compressible and hence memorized. This case suggests that we cannot safely rely on completion as a metric for memorization because it is too strict.

5.5 Conclusion

We introduce a new notion called Compressible Memorization that has advantages over previous notions in several real-world cases. Our definition is a promising direction for future regulation on LLM fair use of data. Future work to explore other ways of measuring compression and faster/cheaper optimization approaches could be fruitful. On a high level, it can be viewed as arithmetic coding with a uniform distribution. Another possible way to compute compression is to compare the negative log-likehood of input versus that of output. One drawback of optimizing hard tokens is the intrinsic hardness of discrete optimization. Instead, if we optimize over the continuous soft token space, the process will

be much more efficient. It remains open how to measure the complexity of a soft token.

5.6 Greedy Coordinate Gradient

Algorithm 2 Greedy Coordinate Gradient (GCG) [Zou et al., 2023]

Input: Loss $\mathcal{L}$, Vocab. $\mathcal{V}$, Target y, Num. Tokens `n_tokens`, Num. Steps `num_steps`
Initialize prompt x to random list of `n_tokens` tokens from $\mathcal{V}$
$E = M$'s embedding matrix
for `num_steps` times **do**
  for $i = 0, \ldots,$ `n_tokens` **do**
    $\mathcal{X}_i = \text{Top-}k(-\nabla_{e_{x_i}} \mathcal{L}(y|x))$
  end for
  for $b = 1, \ldots, B$ **do**
    $\tilde{x}^{(b)} = \mathrm{x}$
    $\tilde{x}_i^{(b)} = \text{Uniform}(\mathcal{X}_i)$, $i = \text{Uniform}([1, \ldots,$ `n_tokens`$])$
  end for
  $x = \tilde{x}^{(b^*)}$ where $b^* = \arg\min_b \mathcal{L}(y|\tilde{x}^{(b)})$
end for
return x

5.7 Random Search

Algorithm 3 Random Search (for LLM prompts) [Andriushchenko, 2023]

Input: Loss $\mathcal{L}$, Vocab. $\mathcal{V}$, Target y, Num. Tokens `n_tokens`, Num. Steps `num_steps`
Initialize prompt x to random list of `n_tokens` tokens from $\mathcal{V}$
for `num_steps` times **do**
  for $b = 1, \ldots, B$ **do**
    $\tilde{x}^{(b)} = \mathrm{x}$
    $\tilde{x}_i^{(b)} = \text{Uniform}(\mathcal{V})$, $i = \text{Uniform}([1, \ldots,$ `n_tokens`$])$
  end for
  $x = \tilde{x}^{(b^*)}$ where $b^* = \arg\min_b \mathcal{L}(y|\tilde{x}^{(b)})$
end for
return x

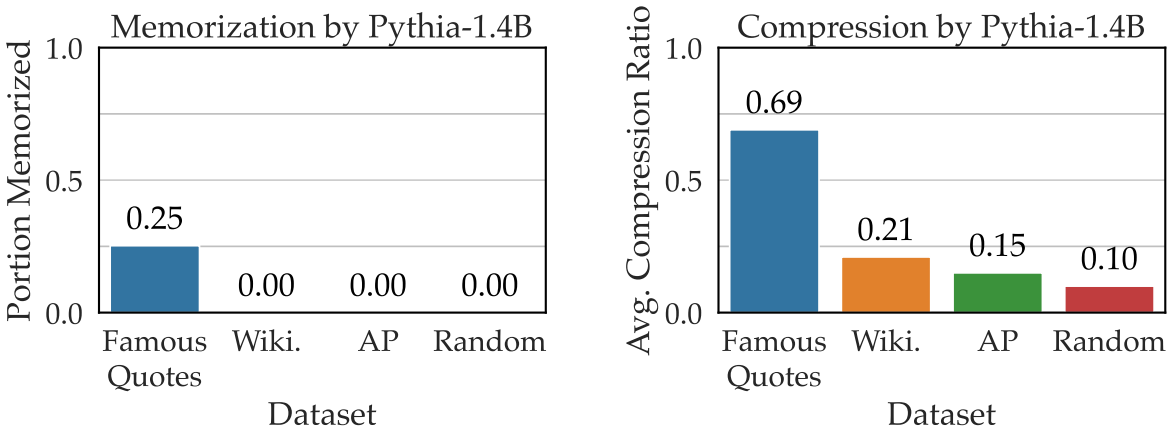


Figure 5.7: Pythia-1.4B Memorization with Random Search.

5.8 More Details of In-context Unlearning

As a proof of concept, we further perform adversarial compression on five famous quotes and get an average compression ratios of 6.54 and 4.74, with versus without the unlearning system prompt, respectively. Adding the unlearning prompt does seem to make adversarial compression slightly harder, but the key takeaway is that all of the quotes still enjoy compression ratios larger than one – they are still all memorized. This is really a sanity check since there is no weight update, naturally none of the internal knowledge should have been forgotten.

Chapter 6

MAGIC: Diffusion Model Memorization Auditing via Generative Image Compression

Based on Gunjan Dhanuka, Sumukh Aithal, Avi Schwarzschild, Zhili Feng, J. Zico Kolter, Zachary C. Lipton, Pratyush Maini. Preprint 2025

Abstract

Diffusion models have revolutionized generative modeling, but their tendency to *memorize* and reproduce specific training images raises significant ethical and legal concerns, particularly regarding copyrighted content. Existing memorization detection methods rely heavily on specific image-text pairs, making them brittle and vulnerable to common prompt modifications used in industry practice. We propose T-MARS(**M**emorization **A**uditing via **G**enerative **I**mage **C**ompression), a novel memorization definition that reframes detection as an image compression problem. The key insight is that if a diffusion model has memorized an image, it should be able to reconstruct that image from a small compressed conditioning, independent of textual prompts. We operationalize this through soft embedding optimization and evaluate compression via multiple criteria including embedding magnitude, token length, and reconstruction quality. Experimentally, T-MARSsignificantly improves robustness and accuracy (by over 20%) compared to existing approaches, maintaining high detection rates even when traditional methods fail under prompt modifications. Our large-scale audit reveals that frontier models memorize over 50% of top copyrighted characters, underscoring the practical importance of robust memorization detection.

Diffusion models have revolutionized generative modeling, but their tendency to *memorize* and reproduce specific training images raises significant ethical and legal concerns, particularly regarding copyrighted content. Existing memorization detection methods rely heavily on specific image-text pairs, making them brittle and vulnerable to common prompt modifications used in industry practice. We propose MAGIC(**M**emorization **A**uditing via **G**enerative **I**mage **C**ompression), a novel memorization definition that

reframes detection as an image compression problem. The key insight is that if a diffusion model has memorized an image, it should be able to reconstruct that image from a small compressed conditioning, independent of textual prompts. We operationalize this through soft embedding optimization and evaluate compression via multiple criteria including embedding magnitude, token length, and reconstruction quality. Experimentally, MAGICsignificantly improves robustness and accuracy (by over 20%) compared to existing approaches, maintaining high detection rates even when traditional methods fail under prompt modifications. Our large-scale audit reveals that frontier models memorize over 50% of top copyrighted characters, underscoring the practical importance of robust memorization detection.

6.1 Introduction

Diffusion models have rapidly emerged as a transformative class of generative models, capable of synthesizing images with striking fidelity and diversity [Ho et al., 2020, Song et al., 2020]. Despite their impressive capabilities, these models' tendency to *memorize* and exactly reproduce specific training images raises significant ethical and legal concerns. Recent lawsuits brought by artists and content creators against generative AI companies highlight the urgency of addressing concerns regarding intellectual property rights and privacy [Sustainable Technology Partners, 2023]. For example, popular text-to-image diffusion models (such as Stable Diffusion, Ideogram, Dall-E 3), trained on massive web-sourced datasets, have been accused of reproducing copyrighted artwork without consent, fueling heated debates around intellectual property infringement and responsible AI development [Somepalli et al., 2023a, Carlini et al., 2023a].

Existing memorization metrics rely heavily on precise text-image pairs or consistent reproduction across multiple generations [Carlini et al., 2023a, Wen et al., 2024]. In practice, these definitions are fragile, easily circumvented by minor modifications to textual prompts during either training or inference. Real-world practices, such as refining or entirely re-captioning training images [Betker et al., 2023, Nguyen et al., 2023], routinely evade detection. Similarly, at inference time, the use of prompt enhancement to improve user prompts has become standard practice. These practices highlight critical limitations of existing auditing methods. To put these claims on solid footing, we empirically analyze the limitations of prior memorization definitions for diffusion models and demonstrate their vulnerability to simple prompt manipulations (Section 6.4).

The key insight behind our approach is that memorization involves *compression*: if a model has memorized specific training content, that information must be compressed and

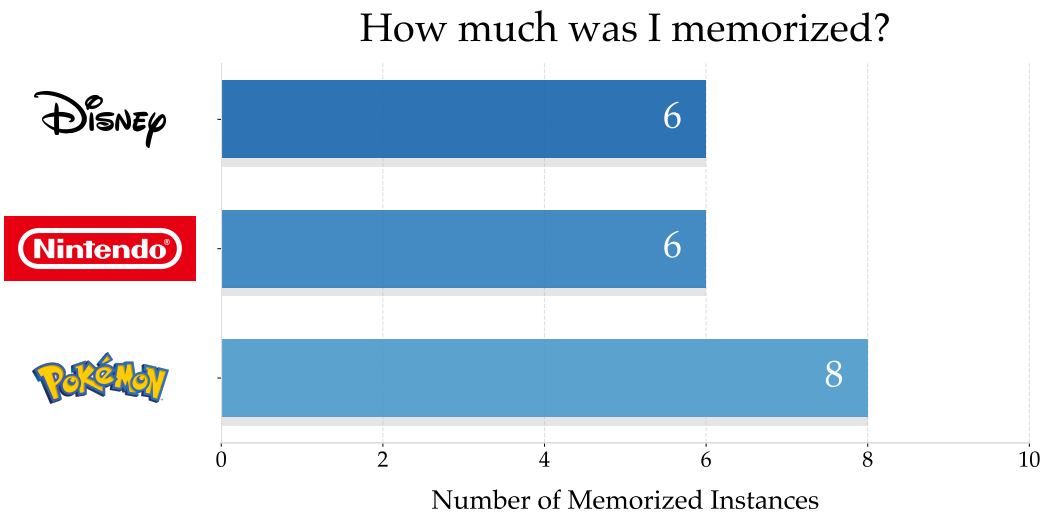


Figure 6.1: How much copyrighted content does SD-2.1 memorize based on the copyright holder? We take the top 10 most famous copyrighted characters from each of three major companies: **Pokémon, Disney, and Nintendo**, and then use MAGICon those images to test whether they were memorized by the diffusion model. We find that over 50% of copyrighted characters were memorized by the model.

stored within the model's learned parameters. This suggests a natural solution—directly probe whether the model can reconstruct specific images using minimal conditioning information, independent of any textual prompts. Such a compression-based test would be inherently robust to textual variations while directly measuring the model's capacity to regenerate memorized content.

Motivated by these insights, we propose MAGIC(**M**emorization **A**uditing via **G**enerative **I**mage **C**ompression), a novel memorization definition that reframes the problem of memorization detection into an image compression task (Section 6.5). MAGICemploys *soft embedding optimization* to compress target images into short, learned embeddings that when used as conditioning, can regenerate the original image through the diffusion model. Intuitively, if a diffusion model has memorized an image, it should easily compress and reconstruct that image from a small embedding, where 'small' may be operationalized based on different measures of compression, as we discuss in Section 6.5. Conversely, images not memorized by the model either cannot be reconstructed faithfully or require substantially 'large' conditioning, effectively distinguishing memorization from generalized learning.

Empirically, MAGICsignificantly enhances robustness and accuracy in detecting memorized content, maintaining high detection rates even when traditional metrics degrade drastically (about 15%) under realistic textual interventions. Our evaluations demonstrate that MAGICreliably identifies memorization overlooked by prior methods, while substantially reducing false positives.

We further leverage MAGICto conduct a comprehensive audit of contemporary diffusion models, focusing on high-stakes copyrighted content. Notably, we uncover that prominent models memorize over 50% of top images from major intellectual property holders, underscoring the practical importance and immediate applicability of our auditing framework. By shifting the discourse from mere output consistency to the deeper question: *can the model regenerate specific training content under plausible conditions?*, we offer practitioners and regulators alike a practical tool for assessing generative models' memorization, ensuring compliance and promoting ethically sound AI deployments.

6.2 Related Work

Memorization in generative models, especially diffusion models, has become a critical area of research due to its implications for privacy, security, and intellectual property rights. This section reviews prior work on analyzing, detecting, and mitigating memorization in diffusion models, as well as technical perspectives on copyright protection in generative AI.

Memorization in Generative Models. Memorization has studied across classes of generative models such as GANs and LLMs. Webster et al. [2019] and others showed that GANs can sometimes overfit and reproduce training examples, proposing metrics to detect such behavior. In NLP, Carlini et al. [2019c] was one of the first to demonstrated that large language models can emit training data verbatim, raising privacy concerns. In fact, a close parallel to our work in NLP is the Adversarial Compression Ratio (ACR) by Schwarzschild et al. [2024a]. Their method marked a string as memorized if could be re-generated by a prompt much smaller than the string itself. However, diffusion models differ substantially in generation process (iterative noising/denoising) and conditioning mechanisms, rendering many existing techniques inapplicable or ineffective. Early evidence of memorization in diffusion models was reported by Somepalli et al. [2023a], who noted that models trained on extremely limited or duplicated data tend to regurgitate specific images. Subsequent work by Carlini et al. [2023a] provided a large-scale confirmation: they extracted dozens of exact training images from Stable Diffusion by cleverly searching the latent space and prompt space. These studies established that diffusion models *do* memorize some training images, especially those that are repeated in the data or have unique features.

Definitions and Detection of Memorization in Diffusion Models. Due to the continuous nature of image generation, defining what constitutes memorization is non-trivial. A straightforward definition is that a model has memorized an image if it can reproduce it

with sufficiently high fidelity (e.g., low perceptual distance) when given the right prompt or latent representation. This definition was operationalized by Carlini et al. [2023a] by searching for prompts that yield images nearly identical to known training examples. A key challenge is setting a similarity threshold — too strict a threshold misses cases of near-memorization, while too lenient a threshold may flag merely similar outputs as memorized. Somepalli et al. [2023b] and Wen et al. [2024] observed that certain rare *prompt phrases* or *trigger words* can consistently cause a model to output the same image, effectively acting as keys to memorized content. Building on this, Wen et al. [2024] proposed an automated test: if multiple generations with different random seeds for a given prompt yield nearly identical images, then the model has memorized that content. Another related field is that of *membership inference*: given an image, determine if it was in the training set [Shokri et al., 2017a, Carlini et al., 2022a]. For diffusion models, the task is harder because the model does not explicitly output training samples unless specifically prompted. [Dubiński et al., 2025] utilize membership signals from diffusion models to perform dataset inference [Maini et al., 2021] for copyrighted content that diffusion models may have been trained on. Our compression-based approach can be seen as a type of membership inference attack specialized for diffusion generative models.

Recently, [Jiang et al., 2025] introduced Inversion-based Inference Perturbation (IIP), a framework for detecting image-level memorization in diffusion models without relying on prompt information. InvMM [Ma et al., 2024] is another inversion-based metric that quantifies image-level memorization in diffusion models by estimating the KL divergence between sensitive latent noise distributions and a standard Gaussian prior. Both IIP and InvMM are computationally intensive, making it less practical for large-scale applications. Another work by Chen et al. [2024a] discussed the concept of local memorization, where only a specific, salient region is replicated instead of the whole image. Another complementary paradigm of detecting memorization in diffusion models is the analytical approach, which seeks to understand memorization by examining the geometric properties of the model's learned probability distribution [Jeon et al., 2025, Brokman et al., 2025]. Chen et al. [2020] proposes definitions for memorization in video diffusion models, and breaks it down into two types: content memorization and motion memorization.

Mitigating Memorization and Content Removal. Several works address how to prevent or limit memorization in generative models. Chen et al. [2024b] proposed strategies to train diffusion models that are less prone to memorizing training data. Data pruning and de-duplication prior to training have been suggested to reduce overfitting on near-duplicate images [Somepalli et al., 2023a, Carlini et al., 2023a]. Recently, techniques for *concept erasure*

in diffusion models have emerged [Kim and Qi, 2025]. Lu et al. [2024a] introduced `MACE`, a finetuning framework that uses targeted LoRA updates to remove the ability to generate specific concepts from a model. CRCE [Xue et al., 2025] uses to identify both semantically related concepts that should be erased alongside the target and distinct concepts that should be preserved. Similarly, Chavhan et al. [2024] explored pruning or editing model weights associated with memorized content, effectively "forgetting" that content without retraining from scratch. [Hintersdorf et al., 2024] localize memorization of images in diffusion models to few neurons in the cross-attention layers and find that removal of these neurons can suppress memorization. These approaches are complementary to ours: while they aim to scrub memorized data from models, our focus is on *detecting and characterizing* memorization.

Ethical and Legal Perspectives. The ability of generative models to reproduce training images verbatim has direct implications for copyright and privacy. Zhang et al. [2023b] provided a technical perspective on copyright protection in generative AI, discussing how models might infringe on intellectual property rights. Vyas et al. [2023] explored provable copyright protection mechanisms for generative models. Heilbut et al. talks about a clear legal analysis of the fair use doctrine applied to Generative Models. A landmark report by U.S. Copyright Office released on Generative AI states that the fair use defense is less likely to succeed if the model is used to generate outputs that "replicate or closely resemble" the training data, thereby harming the market for the original works [United States Copyright Office, 2025]. Our work falls under the broader umbrella of AI model auditing and transparency, sometimes termed the "blue team" approach in AI security: developing tools to ensure models behave responsibly. We note that distinguishing memorization from mere style imitation is an open question: generative models often learn to mimic artistic styles or compositions from training data without copying any single image exactly. This gray area—between permissible inspiration and impermissible plagiarism—remains an area for future work and likely policy intervention.

6.3 Preliminaries

Notation. Let $\mathcal{X} = [0,1]^{H\times W\times 3}$ denote the space of RGB images of fixed resolution and let $\mathcal{Y}$ be the space of text captions. We consider training datasets consisting of image-caption pairs (x, y) where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$. $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$ for a user-chosen noise schedule β_t $\forall t \in \{1 \dots T\}$.

Throughout this paper, we use f to denote the complete generative process (text-to-

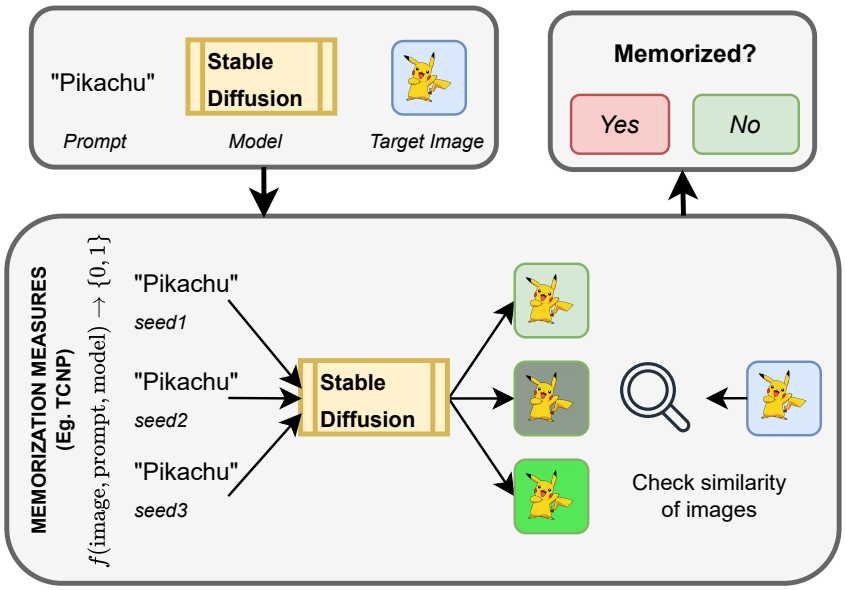


Figure 6.2: An overview of how prompt-based measures for memorization in diffusion models work. The example shown is the TCNP (Text-Conditioned Noise Prediction) metric [Wen et al., 2024], which detects memorization by comparing noise predictions with and without text conditioning. The intuition is that for memorized content, the text prompt heavily influences the denoising process regardless of random initialization, causing large differences between conditioned ($\epsilon_\theta(z_t, t, e(y))$) and unconditioned ($\epsilon_\theta(z_t, t, e(\emptyset))$) noise predictions. When this difference is large across multiple timesteps, it suggests the model has memorized the image-text association.

image), while ϵ_θ specifically refers to the U-Net noise prediction network component. We consistently use x for images in pixel space, z for latents, with subscript t denoting timestep (e.g., z_t for noisy latents). Text embeddings are denoted by $e(y)$ for caption y, with $e(\emptyset)$ for unconditional embeddings. The VAE encoder is denoted $\mathcal{E}$ such that $z_0 = \mathcal{E}(x)$.

6.3.1 Diffusion Models

Diffusion probabilistic models (DDPMs) destroy structure in data through a *forward* noising process and learn to invert it with a neural *reverse* process [Sohl-Dickstein et al., 2015, Ho et al., 2020, Nichol and Dhariwal, 2021]. Modern text-to-image diffusion models like Stable Diffusion consist of several key components that work together to generate images conditioned on text prompts.

Variational Autoencoder (VAE). The VAE consists of an encoder $\mathcal{E}$ that maps images to a lower-dimensional latent space, and a decoder $\mathcal{D}$ that reconstructs images from

latents. For an image x, we have $z = \mathcal{E}(x)$ where $z \in \mathbb{R}^{h \times w \times c}$ with $hL^2(\Omega)H, wL^2(\Omega)W$. The decoder reconstructs via $\hat{x} = \mathcal{D}(z)$. This latent space representation significantly reduces computational requirements while preserving semantic content, and is used downstream in the image generation process.

Text Encoder. Text captions y are processed through a pre-trained text encoder (typically CLIP-style embeddings [Radford et al., 2021b]) to produce text embeddings $e(y) = \texttt{TextEncoder}(y) \in \mathbb{R}^{d \times L}$, where $d = 512$ is the embedding dimension and $L = 77$ is the sequence length for a typical CLIP embedding. These embeddings capture semantic information about the desired image content.

U-Net Denoising Network. The core component is a U-Net ϵ_θ that predicts noise given a noisy latent z_t, timestep t, and text conditioning $e(y)$:

$$\epsilon_\theta(z_t, t, e(y)) \longrightarrow \hat{\epsilon} \in \mathbb{R}^{h \times w \times c}$$

The U-Net incorporates text conditioning through cross-attention mechanisms, allowing the model to generate images that align with the textual description.

Conditioning Mechanism. Text conditioning is integrated into the U-Net through cross-attention layers. At each attention layer, the spatial features of the U-Net serve as queries, while the text embeddings e provide keys and values. This allows different regions of the image to attend to relevant parts of the text description.

Training Objective. Given an image $x \sim p_{\text{data}}$ and its latent $z_0 = \mathcal{E}(x)$, the forward diffusion process progressively corrupts it into a noisy latent z_t.

$$q(z_t \mid z_0) = \mathcal{N}(z_t; \sqrt{\bar{\alpha}_t} z_0, (1 - \bar{\alpha}_t)\mathbf{I}),$$

The model is trained to predict noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ added to clean latents.

$$\mathcal{L} = \mathbb{E}_{t, z_0, \epsilon, y} \left[\| \epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, t, e(y)) \|_2^2 \right]$$

where $e(y) = \texttt{TextEncoder}(y)$ and $\sigma_t^2 = 1 - \bar{\alpha}_t$.

Generation Process. To generate an image from text y, the model starts with random noise $z_T \sim \mathcal{N}(0, \mathbf{I})$ and iteratively denoises using the predicted noise, conditioned on $e(y) =$

$\texttt{TextEncoder}(y)$, ultimately producing z_0 which is decoded to the final image $\hat{x} = \mathcal{D}(z_0)$.

6.3.2 Memorization Definitions

We formalize when a diffusion model f *memorizes* an image-text pair (x, y) by defining memorization tests that return TRUE if memorization is detected.

Definition 5 (Discoverable Memorization [Carlini et al., 2023a]). *Given classifier-free guidance with the original prompt, an image x is $(\texttt{sim}, \tau)$-discoverably memorized if the model f can regenerate it from its original prompt y,*

$$\textsc{Discoverable}_{(\texttt{sim},\tau)}(f, x, y) = \mathbb{1}\left[\texttt{sim}(x, f(y)) > \tau\right],$$

where $\texttt{sim}$ is an image similarity metric, and τ is a threshold. This criterion requires that the known training prompt re-generates the training image.

Image Similarity Metrics As mentioned in Definition 5, discoverable memorization notions require an image similarity metric in order to assess if the generated image resembled the ground truth. We discuss two standard measures of image similarity below that are used in literature.

1. **L2 Distance.** The `L2-Dist` metric considers an image memorized if the pixel-space Euclidean distance between a generated image and a corresponding training image is below a predefined threshold [Carlini et al., 2023a]. Although straightforward, this method is extremely sensitive to minor pixel-level differences and augmentations. For eg., `L2-Dist` would fail if the model generated a mirror image of the training image or if the images have different background colors.

2. **Self-Supervised Copy Detection.** The `SSCD` metric was proposed by Pizzi et al. [2022] as a measure of semantic similarity between two images. This improves upon `L2-Dist` by evaluating perceptual similarity using features extracted from self-supervised models, and has been used in measuring memorization in past works [Somepalli et al., 2023b]. An `SSCD` score greater than 0.7 usually suggests that two images are near perceptual copies of each other. However, SSCD is prone to failing if the image is conditioned to be generated in a different artistic style from the original image as shown by Somepalli et al. [2024].

Definition 6 (Counterfactual Memorization [Wen et al., 2024]). *Let $z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$*

with $z_0 = \mathcal{E}(x)$ *and* $\epsilon \sim \mathcal{N}(0, \mathbf{I})$,

$$\textsc{Counterfactual}_\tau(f, x, y) = \mathbb{1}\Big[\frac{1}{T}\sum_{t=1}^{T}\big\|\epsilon_\theta(z_t, t, e(y)) - \epsilon_\theta(z_t, t, e(\emptyset))\big\|_2 > \tau\Big].$$

Large shifts in noise predictions between text-guided *and* unguided *passes signal that the prompt exerts undue influence, suggesting memorization. Here* $e(y)$ *represents the text embedding of caption* y *and* $e(\emptyset)$ *represents the unconditional embedding.*

Text-conditioned Noise prediction. Wen et al. [2024] examine the magnitude of text-conditional noise predictions, observing that for memorized prompts, the text condition consistently guides the generation toward the memorized image regardless of initializations. Their method achieves high detection accuracy (AUC of 0.960) even at the first generation step with a single generation per prompt, making it significantly more efficient than previous approaches that require multiple generations or querying large training datasets. However, as we show later in the paper, TCNP is dependent on the given prompt being exactly same or similar to the training caption used. Once that condition no longer holds, TCNP suffers a steep performance drop (Figure 6.3).

Relation to traditional counterfactual memorization. Conventionally, the *leave-one-out* definiton of memorization was used to asseess counterfactual memorization [Zhang et al., 2021b], which retrains a model without (x, y) and measures the loss gap. This estimation is computationally prohibitive for modern diffusion systems. TCNP test is an inference time surrogate that assesses the counterfactual scenario when the text-conditioning was not available during generation.

6.4 Brittleness of Existing Memorization Definitions

In this section, we investigate the robustness of existing memorization detection methods for diffusion models, specifically highlighting how they can be easily circumvented through common, practical modifications to textual prompts. We systematically analyze the vulnerability of established metrics under both training-time and inference-time interventions, clearly demonstrating their fragility and motivating the need for a more robust, text-independent detection approach. Figure 6.2 shows how the setup of common prompt-based memorization detection measures.

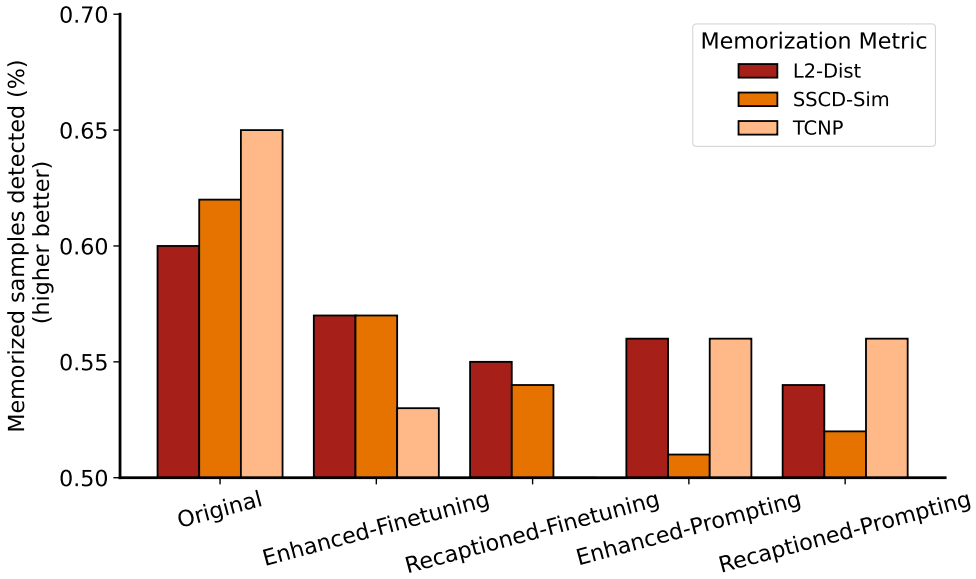


Figure 6.3: Brittleness of existing memorization detection methods under realistic prompt modifications. We evaluate detection accuracy of Discoverable (L2-Distance, SSCD-Similarity), and Counterfactual(TCNP) metrics across five scenarios: (1) **Original**: baseline with exact training captions, (2) **Enh-FT**: training with LLM-enhanced captions, (3) **Recap-FT**: training with VLM-generated captions, (4) **Enh-Prompt**: inference using enhanced prompts, and (5) **Recap-Prompt**: inference using VLM-generated descriptions. All existing methods show significant performance degradation (approaching random guessing at 50%) under these common industry practices, highlighting their fundamental brittleness.

6.4.1 Training-Time Interventions

Training-time prompt modifications are common in modern generative modeling practices. For instance, state-of-the-art models like Stable Diffusion-2 and DALL-E 3 frequently employ synthetic or refined captions to enhance training data quality [Betker et al., 2023]. Such practices inherently challenge existing memorization detection methods that rely heavily on static text-image pairs. To assess this vulnerability, we experiment with two realistic types of training-time prompt modifications:

Training on Enhanced Prompts. We use an LLM (GPT-4o-mini) to generate semantically richer prompts from the original training captions, simulating practical enhancements aimed at clearer, more descriptive training data. By aligning training data with prompt-engineering techniques—such as appending keywords like *high-quality, 4k, ultra-resolution* or imitating specific artistic styles (e.g., camera types or film aesthetics), we aim to make the model more adaptable to diverse user prompts.

Training on Recaptioned Prompts. We simulate a more comprehensive data labeling overhaul by completely recaptioning images with entirely new, descriptive text generated independently via an multimodal LLM. This practice has shown great promise in improving model training [Nguyen et al., 2023, Li et al., 2024c]. To achieve this, we utilize an off-the-shelf VLM, Qwen-2.5 ([Wang et al., 2024b, Bai et al., 2025]), to generate $n = 50$ unique captions per image. The captions are created by posing diverse questions about various aspects of the image, such as its attributes, context, or artistic style (see Appendix for the prompts given to VLM). This process ensures that each image is described from multiple perspectives, resulting in a richer and more varied dataset.

6.4.2 Inference-Time Interventions

We further demonstrate vulnerabilities by evaluating inference-time prompt modifications, which represent typical end-user interactions.

Inference using Enhanced Prompts. Platforms like Ideogram and Midjourney have "Prompt Enhancers" that improve the user's input prompt by appending keywords and detailed explanations to elicit high-quality generations. To simulate this kind of honest inference-time intervention, we use an LLM (GPT-4o-mini) to enhance the original prompt, and pass the enhanced prompt to the Diffusion Model for image generation. The prompt given to the LLM is provided in the Appendix.

Inference using Recaptioned Prompts. If a content-creator/artist wants to check if their image has been memorized by the Diffusion Model, they would need to use a text prompt to work with existing memorization metrics. However, they have no way of knowing what would have been the exact prompt used to train the model. In that case, an effective option is to use a vision-language model to caption the image. To simulate this scenario, we use the Qwen-2.5-VL model to generate a descriptive caption of the image, and use this description as input to the Diffusion Model.

6.4.3 Experiments

To ensure a fair comparison, we reproduced the fine-tuning setup and dataset released by [Somepalli et al., 2023a]. We used Stable Diffusion 2.1 (SD-2.1) [Rombach et al., 2022] as the pre-trained diffusion model and fine-tuned it for 100,000 steps across all experimental setups. This setup simulates a worse-case scenario for adversary where an individual (e.g. content creator, artist) has access to the original captions from the dataset and attempts to determine whether the generated image is memorized by the diffusion model. By using the exact prompts from our 10k subset of the LAION dataset, we assume an ideal case where the individual has access to the precise caption used during training—thereby maximizing the chances of detecting memorization if it occurred.

For clarity, we define the following terminology to refer to the various models used in our experiments: (a) **Original**: SD-2.1 fine-tuned on our 10k subset of LAION using the original image-caption pairs. Use original captions to generate images. (b) **Enhanced-Finetune**: SD-2.1 fine-tuned on our 10k LAION subset, where each caption was enhanced using GPT-4o-mini to provide richer descriptions. Use original captions to generate images. (c) **Recaptioned-Finetune**: SD-2.1 fine-tuned on our 10k LAION subset with 50 diverse captions per image, generated using the Qwen-2.5 vision-language model. Use original captions to generate images. (d) **Enhanced-Prompting**: SD-2.1 fine-tuned on our 10k LAION subset using the original image-caption pairs. Use LLM-Enhanced captions to generate images. (e) **Recaptioned-Prompting**: SD-2.1 fine-tuned on our 10k LAION subset using the original image-caption pairs. Use descriptions generated by a VLM to generate images.

To construct the memorized subset, we select the 100 samples with the highest SSCD similarity scores between the generated images and their corresponding originals. Visual inspection of these samples confirms strong resemblance, suggesting likely memorization. In contrast, the non-memorized subset comprises the 100 samples with the lowest similarity scores. We then demonstrate the brittleness of these memorization-based classifications by

applying training-time and inference-time interventions to these selected subsets.

Figure 6.3 shows that simple interventions to the prompts dramatically reduce memorization detection accuracy across all metrics. For the state-of-the-art TCNP metric, the detection accuracy plummets from 65% to 50% (random guessing) in the worst case. Performance suffers for L2-Dist and SSCD-Similarity based classification as well and the values after interventions are all close to 50%, which essentially implies random guessing. It is noteworthy that the interventions are not strictly adversarial in nature, and can be made by an honest model developer to improve the performance of their Diffusion Models.

6.4.4 Implications and Motivation for New Metrics

The experiments clearly demonstrate that memorization detection methods relying on fixed textual prompts are fundamentally brittle and ineffective in realistic, practical settings. Minor and natural textual variations, common in both training data refinement and end-user interactions, lead to substantial detection failures, underscoring a critical weakness of existing metrics.

This motivates the urgent need for memorization detection methods independent of textual inputs, directly probing intrinsic model capabilities. We address this gap in the following section by introducing a robust, compression-based auditing method that circumvents these textual vulnerabilities.

6.5 MAGIC: Memorization Auditing by Generative Image Compression

6.5.1 Method Overview

Given the shortcomings of existing memorization metrics (Section 6.4), a robust approach to auditing memorization must satisfy two critical properties: (1) **independence from the text prompt**, requiring only the image itself, and (2) **direct probing** of the model's learned visual representation rather than its entire generative pipeline. Motivated by these insights, we propose MAGIC(**M**emorization **A**uditing via **G**enerative **I**mage **C**ompression), which frames memorization detection as an image compression task. Intuitively, if a model has memorized an image, it should have *compressed* information about it in the visual encoder, and hence should be able to *reconstruct* it using minimal conditioning information. In addition to being independent of the text prompt, and allowing for direct probing, MAGICadditionally allows for a **quantitative measurement** of memorization. In particular,

we can measure 'how compressed' the image was based on the size of the conditioning, as opposed to providing for a binary measure of memorization.

Definition 7 (**Compression-based Memorization** (MAGIC))**.** *Given a compression function c that maps an image x to a minimal conditioning representation, and a threshold τ:*

$$MAGIC_{\tau,c}(f, x, y) = \mathbb{1}\left[c(f, x) < \tau\right],$$

where $c(f, x)$ quantifies the minimal conditioning information required for model f to reconstruct image x with high fidelity. The compression function c can be operationalized in multiple ways (e.g., embedding magnitude, token length, reconstruction quality) as detailed in Section 6.5.3. MAGICfinds an optimal continuous embedding (conditioning vector) for each target image by minimizing the model's reconstruction error on that image. This embedding can be thought of as a "compressed representation" for the image within the model's learned space. Unlike existing text-dependent definitions, MAGICdirectly probes the model's learned visual representation through optimized embeddings, making it robust to textual variations. The closest analogue to MAGICis Adversarial Compression Ratio (ACR) [Schwarzschild et al., 2024a], a compression-based definition of memorization in LLMs.

6.5.2 Soft Embedding Optimization for Image Compression

The core idea behind MAGICis optimizing a short, continuous conditioning embedding e to reconstruct a given target image x. Specifically, we find e by minimizing reconstruction loss over the diffusion process. Let $z_0 = \mathcal{E}(x)$ represent the VAE-encoded image latent. Starting from a generic embedding (e.g., encoding "an image"), we iteratively update e to minimize the standard diffusion training loss:

$$L(e) = \mathbb{E}_{t,\epsilon}\left[\frac{1}{2\sigma_t^2}\|\hat{\epsilon}_\theta(z_t, t, e) - \epsilon\|_2^2\right], \tag{6.1}$$

where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, $t \sim \text{Uniform}(1, T)$, $z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$, and $\sigma_t^2 = 1 - \bar{\alpha}_t$. The full procedure is detailed in Algorithm 4. Crucially, only e is optimized, while the diffusion model weights remain unchanged.

Normalization via JPEG Compression. To account for the intrinsic complexity of images (i.e., some images inherently compress better due to their simplicity), we normalize all our embedding metrics relative to JPEG compression size, a "zero knowledge" baseline as introduced by Somepalli et al. [2023b]. This normalization helps control for the inherent

Algorithm 4 Memorization Auditing via Generative Image Compression (MAGIC)

Input: Target image x, diffusion model ϵ_θ, VAE encoder $\mathcal{E}$, initial embedding e, learning rate η, iterations N
Encode image: $z_0 \leftarrow \mathcal{E}(x)$
for $i = 1$ to N **do**
    Sample timestep $t \sim \text{Uniform}(1, \ldots, T)$
    Sample noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
    Create noised latent: $z_t \leftarrow \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon$
    Predict noise: $\hat{\epsilon} \leftarrow \epsilon_\theta(z_t, t, e)$
    Compute loss: $L \leftarrow \frac{1}{2\sigma_t^2}\|\hat{\epsilon} - \epsilon\|_2^2$
    Update embedding: $e \leftarrow e - \eta \nabla_e L$
end for
return Optimized embedding e

compressibility of an image, thus isolating the memorization signal more accurately.

The Importance of Initialization. In order to converge to the solution fast, we notice that initializing the text guidance with the actual prompt of the image can be extremely beneficial. In particular, by assuming access to either the (i) original text associated with caption; or (ii) a VLM generated caption of an image, we can significantly speed up the optimization process. This observation both points to a practical strategy to yield the best results out of MAGIC, but also an important limitation of the optimization process itself, which would be of interest for future work.

6.5.3 Operationalizing MAGIC

We test three variants of our memorization definition, each leveraging a distinct property of the optimized embedding. These variants can be thought of as ways to operationalize the compression-based definition proposed in this paper. Similar to how discoverable memorization Definition 5 requires a similarity metric (such as L2-norm or SSCD), MAGI-Cutilizes various forms of compression to operationalize the definition.

The three variants capture complementary aspects of compression: **embedding magnitude** measures how much conditioning information is needed (smaller norms suggest easier reconstruction), **token length** directly measures compression efficiency (fewer tokens indicate better memorization), and **reconstruction quality** measures how faithfully the image can be recovered (higher similarity suggests stronger memorization). Together, these provide a comprehensive view of memorization from different compression perspectives.

Embedding Norm (L2 Norm). This variant quantifies memorization by measuring

the Frobenius norm $|e|_F$ of the optimized embedding, normalized by JPEG compression size. Intuitively, a lower embedding norm indicates the model requires less information to reconstruct the image, suggesting stronger memorization. To ensure that the embedding norm is minimized during the optimization process, we apply L2 weight decay regularization during embedding optimization.

Token Length (Compression Factor). Each embedding e in Stable Diffusion has dimensions $512 \times N_{\text{tokens}}$ (e.g., 77 tokens). In this variant, we search for the minimum number of tokens needed to achieve faithful image reconstruction (SSCD score > 0.7, which corresponds to the threshold τ in our definition for this variant). A smaller required token count indicates stronger memorization. We also normalize this metric relative to JPEG compression size. The search for minimum token count is based on a grid search among the following values of tokens and stopping at the smallest value that satisfies the SSCD threshold: $\{4, 8, 16, 30, 40, 50, 60, 70, 77\}$.

Reconstruction Similarity (SSCD). This metric represents the maximum achievable similarity (as measured by SSCD) between the original and the reconstructed image, given the full embedding dimension. A high SSCD score indicates successful reconstruction and is therefore a direct indicator of memorization capability.

Combined Metric (All). We also consider a combined approach (MAGIC*All*), where we jointly leverage embedding norm, token length, and reconstruction similarity through a logistic regression model trained on a validation set to predict memorization.

6.5.4 Experiments and Results

In this section, we present a thorough evaluation of MAGIC, clearly defining the metrics used, their intuitive meaning, and the reasoning behind each variant. We also provide a comparison against existing state-of-the-art memorization detection methods.

Setup. The evaluation setup for this section extends from the discussion about the limitations of existing memorization metrics. We follow established experimental setups from recent literature [Carlini et al., 2023a, Somepalli et al., 2023b, Wen et al., 2024] but use the more recent Stable Diffusion 2.1 model, which has also been trained on de-duplicated dataset, making finding memorized images even tougher. We fine-tune a Stable Diffusion 2.1 model on 10k image-caption pairs sampled from the LAION dataset. We then choose the 100 most memorized images based on SSCD score of the original and generated image. Since we use SSCD to find the examples, we use a fixed threshold of 0.5 (corresponding to τ in Definition 5) to report its accuracy, as opposed to finding the threshold at 5% FPR.

Results in Table 6.3 indicate robustness to both training and inference-time interventions.

Parameter	Value
Number of optimization iterations	1000
Learning rate	0.01
L2 weight decay	0.001
Batch size	1 (one image at a time)
Embedding initialization	Random / CLIP encoding of image
Optimizer	AdamW [Loshchilov and Hutter, 2017]
Diffusion model	Stable Diffusion 2.1 [Rombach et al., 2022]
Number of tokens	$\{4, 8, 16, 30, 40, 50, 60, 70, 77\}$
Embedding size	(Number of tokens) $\times$ 1024
VAE	From diffusion model
Guidance	7.5
Timestep sampling	$\mathcal{U}(50, 950)$

Table 6.1: Hyperparameters and experimental configuration for MAGIC.

MAGICmaintains high accuracy and robustness, significantly outperforming prior caption-dependent definitions, and their operationalizations. The MAGICdefinition fundamentally shifts memorization auditing from a reliance on fixed prompts towards directly probing model capabilities. This offers (i) **Robustness to prompt variability**: Independent of textual conditioning, making it resilient to common real-world modifications (Section 6.4); (ii) **Practical feasibility**: No requirement to access original training captions or large datasets—only the diffusion model and the target image.

6.6 How much copyrighted content do frontier models memorize?

Having redefined memorization detection through generative image compression (Section 6.5), we now leverage our approach to empirically audit memorization in frontier diffusion models, focusing on popular copyrighted characters. Unlike prior methods, which depend heavily on static image-caption pairs and fail to reliably detect memorization under minor textual perturbations (Section 6.4), MAGICrobustly identifies memorization purely based on image content. Using the thresholds established in Section 6.5.4 (corresponding to a 5% FPR), we perform a rigorous evaluation on a systematically curated set of iconic images representing major intellectual properties.

Specifically, we evaluate ten popular and recognizable characters or images from each of the following major copyright holders: **Pokémon, Disney**, and **Nintendo**. Images are selected using the top Google search result for each character (as of April 2025),

Method	Runtimes	Description
DISCOVERABLE$_{\texttt{(L2-Dist)}}$	13.09 ± 0.11 s	forward pass through model and distance computation
DISCOVERABLE$_{\texttt{(SSCD)}}$	14.12 ± 0.05 s	passing through SSCD model backbone
TCNP	14.18 ± 0.16 s	one forward pass through Diffusion model
MAGIC	230.27 ± 76.41 s	1000 optimization steps × ~ 0.27 ms/step
IIP	–	could not reproduce, code missing [latest commit]
InvMM	1.5 hours	for 2000 training steps

Table 6.2: Runtime Comparison for Different Memorization Detection Methods. We compare the time taken to detect memorization of MAGICagainst existing memorization detection methods, including DISCOVERABLE$_{\texttt{(L2-Dist)}}$ [Carlini et al., 2023a], DISCOVERABLE$_{\texttt{(SSCD)}}$ [Somepalli et al., 2023b], TCNP [Wen et al., 2024], IIP [Jiang et al., 2025] and InvMM [Ma et al., 2024]. In each run, we compute the memorization result for one target image on a single NVIDIA RTX A6000 GPU. Then (except for IIP and InvMM), we compute the mean and standard deviation across 100 such runs (with unique target images).

simulating plausible and common scenarios of memorization in publicly-trained models. (See Appendix for full list.)

Figure 6.1 summarizes our findings. We observe substantial memorization across all tested copyright holders. All three companies show significant memorization, with over 50% of characters memorized across the dataset.

Importantly, our evaluation reveals that certain images, like iconic depictions of characters (e.g., Mickey Mouse or Pikachu), have been memorized multiple times under various different captions. This insight highlights a crucial limitation of prior caption-dependent metrics [Wen et al., 2024, Carlini et al., 2023a], which cannot effectively handle such real-world memorization scenarios.

This analysis underscores the pressing need for robust memorization auditing tools, particularly in light of ongoing lawsuits, such as the well-known litigation against Stability AI by artists and corporations alleging unauthorized reproduction of protected content [Sustainable Technology Partners, 2023]. By quantifying memorization concretely and independently from textual conditioning, MAGICprovides valuable transparency for regulatory scrutiny, legal contexts, and ethical development of generative AI technologies.

6.7 Conclusion

In this paper, we introduced MAGIC, a novel memorization definition for auditing memorization in diffusion models via generative image compression. By framing memorization detection as an image compression problem, MAGICovercomes critical limitations of exist-

Method	Original	Enh-FT	Recap-FT	Enh-Prompt	Recap-Prompt
DISCOVERABLE$_{(\text{SSCD},0.5)}$, $n = 4$	0.62	0.57	0.54	0.51	0.52
	Accuracy (↑) at 5% FPR Threshold on Original				
DISCOVERABLE$_{(\text{L2},0.5)}$, $n = 4$	0.60	0.57	0.55	0.56	0.54
TCNP, $n = 4$	0.65	0.53	0.50	0.56	0.56
MAGIC(w/ L2 Norm)	0.73	0.72	0.52	0.73	0.73
MAGIC(w/ Token Length)	0.79	0.80	0.84	0.79	0.79
MAGIC(w/ Recon. Sim.)	0.81	0.79	0.73	0.81	0.81
MAGIC(All)	0.84	0.81	0.85	0.84	0.84

Table 6.3: Memorization Detection Results Across Metrics and Interventions. We compare the performance of MAGICagainst existing memorization detection methods, including Density$_{\ell_2}$ [Carlini et al., 2023a], Density$_{\text{SSCD}}$ [Somepalli et al., 2023b], and TCNP [Wen et al., 2024]. Each column corresponds to a variant of the diffusion model (original or subjected to training-time or inference-time prompt modifications). The first row uses a fixed threshold (SSCD=0.5), while subsequent rows report accuracy at the threshold corresponding to 5% false positive rate (FPR) on the original model. All baseline methods rely on fixed prompts, whereas MAGICuses image-only compression. Darker green indicates higher detection performance (closer to 1.0), while red/yellow indicates performance close to random guessing.

ing definitions that rely heavily on textual conditioning and are thus vulnerable to practical training and inference-time interventions. MAGICleverages soft embedding optimization to identify if and how well diffusion models can reconstruct specific images solely based on learned continuous embeddings.

Empirically, meauring memorization via compression-based methods significantly improves robustness and accuracy in detecting memorized content, maintaining reliable detection performance even when methods operationalizing existing definitions degrade sharply under realistic textual variations. The comprehensive evaluations presented in this work demonstrate that MAGICnot only accurately identifies memorized images that traditional metrics overlook, but also substantially reduces false positives. Thus, MAGICprovides practitioners and regulators with an effective, scalable, and reliable tool for auditing memorization in generative models.

Furthermore, our large-scale audit based on MAGICrevealed significant memorization in frontier diffusion models, especially with regards to high-value copyrighted content. By highlighting these risks quantitatively and transparently, our memorization definition, and its operationalization, underscore the importance of rigorous memorization auditing to ensure ethical compliance and mitigate legal and societal harms.

Figure 6.4: Increasing the number of tokens increases the reconstruction fidelity in the MAGIC Soft Prompt Optimization. However, after N=4, the additional tokens do not capture significant extra information and are sparse in nature. Thus, it can be said that N=4 tokens is sufficient to generate a near-exact copy of a Pikachu image.

Future Work and Limitations Several promising directions arise from our work. First, extending MAGICto other generative modeling paradigms, such as video (like SORA) or 3D generative models, is an immediate and compelling avenue. Given the increasing prominence and potential privacy implications of these modalities, adapting our compression-based auditing framework could significantly advance the understanding and control of memorization across diverse media types. Second, our memorization definition currently focuses strictly on exact or near-exact image reconstructions. This specificity may miss nuanced forms of memorization, such as semantic or stylistic memorization, which may also carry significant ethical implications. Examining partial or semantic-level memorization remains a critical next step. Third, MAGICstill requires iterative optimization, especially if initialized poorly or without prior textual guidance. Future work optimizing this process could make the optimization process more reliable and fast.

Chapter 7

MemSinks: Memorization-Aware Learning via Controlled Sinks

Based on Gaurav R. Ghosal, Pratyush Maini, Aditi Raghunathan. ICML 2025 (under review)

Abstract

Large language models are susceptible to memorizing repeated sequences, posing privacy and copyright concerns. A popular mitigation strategy is to remove memorized information from specific neurons post-hoc. However, such approaches have shown limited success so far. In a controlled setting, we show that the memorization of *natural* sequences (those that resemble linguistically plausible text) become *mechanistically entangled* with general language abilities, thereby becoming challenging to remove post-hoc. In this work, we put forward a new paradigm of MemSinksthat promotes isolation of memorization by design. We leverage a sequence identifier that activates a unique set of memorization neurons for each sequence across repetitions. By analyzing the dynamics of learning and forgetting, we argue that MemSinksfacilitates isolation of memorized content, making it easier to remove without compromising general language capabilities. We implement MemSinksat the billion-parameter and billion-token scale, and observe both effective isolation and strong generalization. To our knowledge, this is the first proof-of-concept on real data demonstrating that simultaneous generalization and isolation is achievable. We open-source our code at `http://github.com/grghosal/MemSinks`.

7.1 Introduction

Large language models often memorize sequences that are repeated during pretraining [Carlini et al., 2023d, Nasr et al., 2023b], posing concerns in privacy, copyright, and membership inference. There is significant research on the problem of "unlearning" or removing memorized information from models post-hoc, including fine-tuning based approaches [Maini et al., 2022, Barbulescu and Triantafillou, 2024]. However, presently all such methods present a substantial tradeoff between removing memorization and

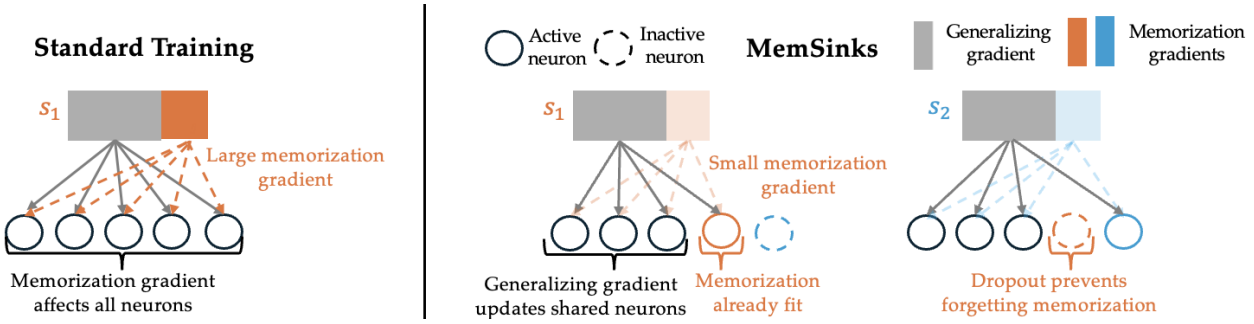


Figure 7.1: Conceptual Intuition of MemSinks. For illustration, we assume that gradient of each sequence can be decomposed into a generalizing (gray) and memorizing component. Under standard training (shown in left) memorization can be arbitrarily distributed throughout the model. In MemSinks, on the other hand, memorization sink neurons are protected from forgetting and as a result accumulate memorization in a known subset of neurons (illustrated on the right hand side).

preserving general model capability.

We identify a core reason for the tradeoff: standard training induces *mechanistic entanglement*, where the same components support both generalization and memorization [Scherlis et al., 2025]. This occurs when memorization relies on mechanisms also used for general language understanding. In a controlled setting, we show that repeated natural text sequences are memorized with such entanglement, making it challenging to remove them without harming general performance. We further prove that gradient descent has an implicit bias toward such entangled solutions, suggesting that **mechanistic entanglement is inherent to current training methods.**

Can new training approaches better disentangle memorization and general language capabilities?

We begin with a natural disentanglement attempt (Section 7.4; see also [Cloud et al., 2024]) by restricting gradient updates from repeated sequences to designated "memorized components", while the remaining "general components" learn only from non-repeated data. This approach has two major flaws. First, it weakens generalization by depriving general components of all training signal from potentially high-quality repeated sequences. More subtly, generalization further degrades when memorized components are removed at inference: the model's general capabilities co-adapt to the memorization neurons during training. Removing them post-hoc breaks this dependence and harms performance.

Drawing lessons from the failures above, we introduce Memorization Sinks (MemSinks) — inspired by the previous study of localization in Maini et al. [2023].

Memorization Sinks are neurons that are consistently and selectively activated when training on a sequence (and dropped out on other sequences). As a result, these sinks accumulate sequence-specific memorization and can be dropped out to remove it.

Note that this approach addresses the implicit dependence issue identified earlier, and also allows the shared components to learn from repeated sequences. But what prevents the shared components from containing memorized information? Does it suffer from the same mechanistic entanglement issue of standard training?

The key insight lies in the difference in the training dynamics of generalizing and memorization signal, which has been studied in disparate lines of work. Generalizing signal is consistently amplified across training sequences [Chatterjee, 2020]. However, memorization signals of different training sequences interfere with each other, resulting in a cyclical learning-forgetting dynamic [Toneva et al., 2018]: learning when the repeated sequence is seen, but forgetting when training on a different sequence due to interference. In standard training, there is no separation of model components so this learning-forgetting dynamic occurs throughout the model leading to mechanistic entanglement. However, in MemSinks, this cycle is broken by allocating separate components per repeated sequence that are **protected from interference** with other sequences. As a result, memorized information is not reinforced in the shared components across repetitions of sequences.

How well does MemSinksdisentangle memorization and generalization in practice? We train 360M and 1.7B SmolLM models [Allal et al., 2025] on the SlimPajama dataset [Shen et al., 2024], with repeated sequences drawn from TinyStories. Standard training leads to strong memorization where repeated sequences show significantly (i.e. 200x) lower loss than held-out ones. With MemSinks, dropping the memorization components closes over 50% of this loss gap, mitigating memorization. Furthermore, MemSinks(without memorization components) matches the validation loss of standard training and significantly outperforms a de-duplication baseline, preserving the benefits of repeated data for generalization. This provides a proof-of-concept that MemSinkscan disentangle memorization from generalization in realistic settings (Section 7.2.2).

We perform various ablations on the impact of design choices for implementing MemSinkson a small-scale TinyStories pre-training task in Sections 7.2.1 and 7.2.3. To assess the practicality of MemSinks, we investigate two key axes: (1) scalability with respect to model and data size, and (2) robustness to noise in sequence ID assignment. While our proof-of-concept assumes access to perfect sequence IDs, real-world deployments would require tolerance to imperfect or noisy annotations. We find that MemSinksis robust to small levels of noise in the sequence IDs (up to 10%) and works across a range of model

sizes. Importantly, the benefits of MemSinksscale with increasing model size, suggesting promise at larger scales.

In summary, we argue that post-hoc unlearning on standard trained models is fundamentally limited due to inevitable mechanistic entanglement between memorization and generalization components. We propose a new training paradigm, MemSinks, and provide a conceptual explanation for how it enables clean disentanglement. Empirically, we show that MemSinkssupports post-hoc removal of memorized content without compromising performance on real-world data. We also demonstrate its practicality: the approach scales well, with gains preserved, and even amplified, as we scale up models and datasets, and it remains robust under noisy sequence ID metadata. Ultimately, MemSinksoffers a concrete path forward for the challenge of removing memorization without degrading model capabilities.

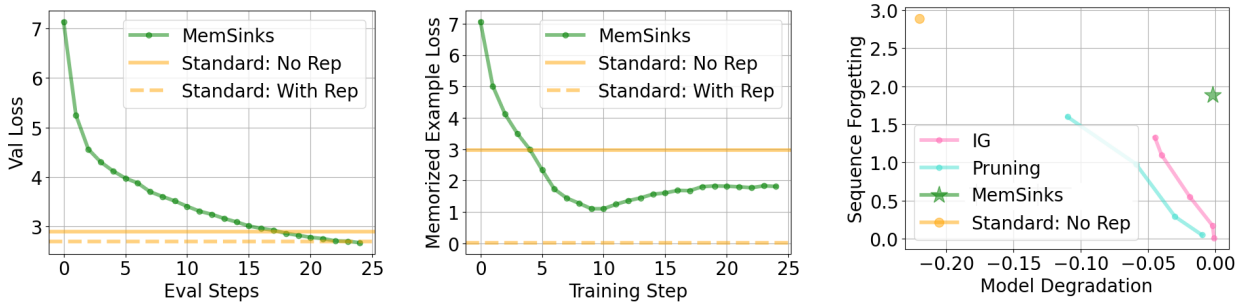
7.2 Channeling Training Dynamics with MemSinks

In Section 7.4, we found that rigidly partitioning gradients to achieve disentanglement is insufficient. Our result suggest the need to enforce a more precise separation between memorization and general capability. However, it is challenging to exactly specify what consitutes memorization versus generalization during the training process.

To address this, we leverage the difference in training dynamics of memorization and generalization. During training, generalizing signals are reinforced across distinct sequences and are broadly amplified [Chatterjee, 2020]. Sequence-specific memorization, on the other hand, experiences interference from other examples and is gradually forgotten. As a result, repeated sequences are cyclically learned and forgotten throughout the training process, thereby becoming defused throughout the model.

We propose to allow memorization to build up in pre-specified *memorization sink* neurons by shielding them from interfering updates. Intuitively, by setting aside a known and stable location for storing memorization, we reduce the need for it to be reinforced across all model parameters. After training, removal of memorization can be achieved by simply deleting the memorization sinks.

Our design requires two properties of the memorization sink neurons: (a) they must experience less interference from unrelated sequences and (b) they must avoid *co-adaptation* with the rest of the model (as discussed in Section 7.4). We propose that both objectives can be achieved through *sequence-dependent dropout*. Specifically, a different subset of sink neurons is activated for each input sequence. This selective activation reduces the frequency of interfering gradient updates to sink neurons, shielding them from forgetting. At the same time, their infrequent activation also regularizes the rest of the model against co-adaptation.



(a) Validation Loss **(b)** Loss on Memorized Examples **(c)** Unlearning-Model Degradation Tradeoff

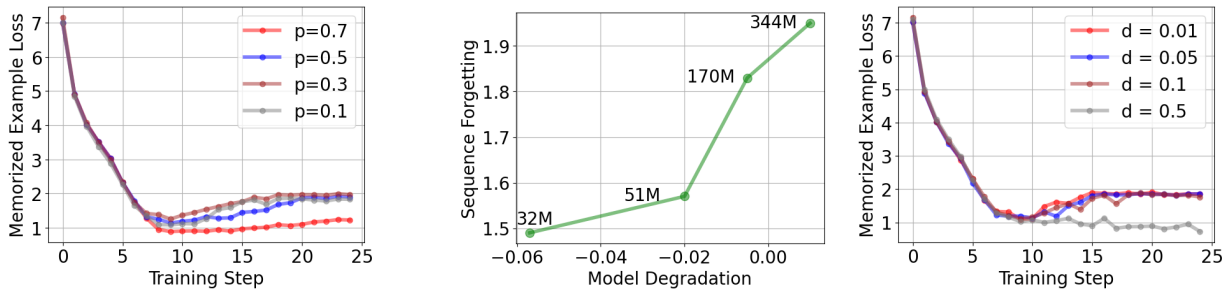
Figure 7.2: Performance of MemSinks (a) We find that MemSinksachieves a comparable validation loss to a normally trained model on `repeat`, outperforming a model trained without repeated sequences. (b) We show the loss of MemSinkson the repeated sequences, showing that it memorizes significantly less than a normally trained model. (c) We compare the sequence forgetting-model degradation tradeoff of MemSinks, relative to the post-hoc methods tested in Section 7.3, finding MemSinksoutperforms both. We compute the model degradation for MemSinksand Standard: No Rep as the difference in the validation loss relative to a standard trained model on `repeat`.

Implementation We implement MemSinksin the hidden layer of transformer MLPs, as prior works have found MLPs play a crucial role in storing memorization [Nanda et al., 2023, Geva et al., 2021]. We set a fraction g of the hidden layer neurons in each layer as generalization neurons (which are activated across all sequences), while the remaining $1-g$ fraction serve as memorization sink neurons. We assign each sequence in the pretraining dataset an ID and mask the memorization sink neurons deterministically as a function of the sequence ID. During all evaluations, we remove memorization sink neurons. Further implementation details are provided in Appendix 7.4.3.

Experimental Details We train a GPT Medium model (same as all previous experiments), where 70% of MLP neurons are shared and the remaining 30% are allocated to the pool of memorization neurons. We emphasize that there are *far less* memorization neurons than total sequences. Thus, we do not assume each sequence can be allocated its own memorization neurons. We set the memorization neuron dropout ratio p to 0.3, but explore other choices in Section 7.2.3. We train on the `repeat`dataset from Section 7.3.

7.2.1 Validation of MemSinksin TinyStories

MemSinksEnable Learning Across Sequences In Figure 7.2a, we compare the validation loss of MemSinkswith standard training with and without repeated documents. Firstly, note that standard training with repeated sequences outperforms filtering them out. This indicates that the model does learn general capabilities from observing documents repeated



(a) Impact of Activation Ratio

(b) Impact of Model Size

(c) Robustness to Sequence ID Noise

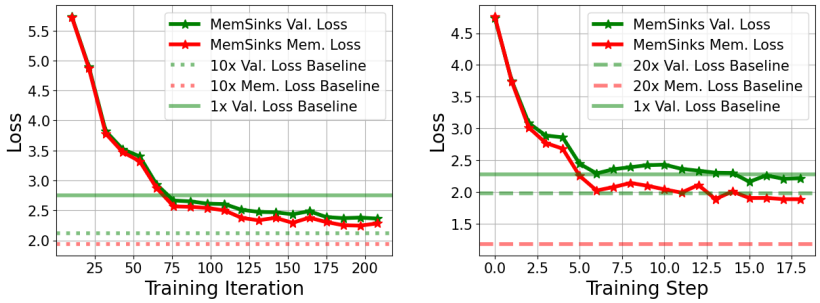
Figure 7.3: Practicality of MemSinks (a) We study the impact of the fraction of memorization neurons activated (p) on any given sequence. We find that the model is generally robust to this choice, but activating too many can interfere in the isolation of memorization. (b) We plot the tradeoff between removing memorization and model degradation across model sizes, finding that larger models achieve a better tradeoff. (c) We study the impact sequence ID noise d, where a fraction of repeated documents have an inconsistent ID. We find that MemSinkswithstands small amounts of noise (up to 10%).

multiple times in our setting. Next, we compare the standard trained models with MemSinks. We observe that (evaluating without the memorization neurons), MemSinksachieves comparable validation loss to standard training.

Dropping Out MemSinksRemoves Memorization In Figure 7.2b, we show the loss on the repeated TinyStories documents. A standard trained model memorizes these sequences during training, achieving close to 0 loss on them. We observe that dropping out the memorization neurons significantly increases the loss on these sequences, increasing the loss to roughly 66% of a standard trained model that does not memorize. Interestingly, we observe that in the latter part of training, the loss of sequence-tied dropout on memorized sequences begins to increase. This suggests that while shared neurons may initially implement some memorization, further training reverses this.

7.2.2 Larger Scale Experiments with MemSinks

In the previous section we studied MemSinksin a small scale setting. Now, we provide empirical validation of MemSinkson a larger scale, involving open pretraining datasets. We concentrate on settings where repeating data is beneficial for generalization (i.e. by upsampling a rare domain), but we also wish to avoid exactly memorizing the specific repeated documents.



(a) 360M Parameters **(b)** 1.7B Parameters

Figure 7.4: Larger Scale Experiments on MemSinks We plot the validation and memorization losses of (a) 360M and (b) 1.7B SmolLM-style models on a mixture of SlimPajama and TinyStories data. We compare with standard training with and without repeated TinyStories data. Our results demonstrate that across both settings, MemSinksis able to mitigate memorization (relative to standard training) without harming validation loss (outperforming or comparable to deduplicated training).

Setting We focus on pretraining SmolLM 360 and 1.7 billion models on 1 billion and 2 billion tokens, respectively. The majority of the pretraining dataset consists of a subset of the SlimPajama dataset, with a small amount being sampled from TinyStories. Our goal is to simulataneously improve performance on validation TinyStories data while also avoiding memorization of the repeated TinyStories sequences. Full details are presented in the Appendix 7.4.3.

MemSinksPreserve Benefits of Repeated Data In the settings we study, validation performance on TinyStories benefits from repetition (by upweighting this relatively under-represented distribution). This can be seen in Figure 7.4a in the validation loss gap between the *deduplication* baseline (where the TinyStories are seen only once) and the full repetition baseline. We observe that the validation loss attained by MemSinksoutperforms the deduplication baseline (achieving validation loss comparable to the repeated baseline). This provides evidence that MemSinkscan leverage the generalization benefits of repeated data.

MemSinksMitigates Memorization Although repeating data has benefits for generalization, it also results in memorization – as evidenced by the gap between the validation loss and the loss on training examples. However, as shown in Figure 7.2b we find that training with MemSinksachieves a significantly higher loss on the training data, substantially closing the gap between validation and training loss (by at least 50%). In a setting with more extensive repetition (20x), we additionally find that the mitigation of memorization by MemSinksis more pronounced.

Collectively, our findings in this section provide promising proof-of-concept evidence

that MemSinkscan effectively separate memorization and generalization in the presence of heterogeneous and noisy data that constitute the real-world LLM pretraining corpora.

7.2.3 Practicality of MemSinks

In this section, we examine the robustness of MemSinksto various hyperparameters and variation in the training process. We focus on the loss on memorization examples as a measure of MemSinks's success in isolating memorization. We perform the experiments here in the small-scale TinyStories setting from Section 7.2.1.

Impact of MemSinksActivation Ratio A key hyperparameter for MemSinksis the fraction of memorization neurons activated on any given sequence (p). This controls across how many documents a particular memorization neuron is activated, as well as how much memorization capacity is allocated to a given sequence. We see that MemSinksis generally robust to this parameter, with a relatively small value $p = 0.3$ performing best. The performance of MemSinksdoes break down at higher values (i.e. $p = 0.7$). We hypothesize this breakdown arises as a result of insufficient shielding of memorization sinks from forgetting and further investigate this in Section 7.2.4.

Impact of Model Size Another concern is that MemSinkscould necessitate using significantly larger models. In Figure 7.3b, we test the performance of MemSinkson a range of model sizes and find that it is capable of isolating memorization across model scales—as indicated by the comparably high losses on repeated sequences (relative to a normally trained model which attains nearly 0 loss). On the other hand, we find that model degradation (the increase in validation loss compared to a standard trained model of the same size) does grow as the model architecture becomes smaller. However, even on smaller models MemSinksoutperforms post-hoc methods as shown in Figure 7.2c. Thus, while model size plays a role in the success of MemSinks, the method has benefits in small models as well.

Robustness to Masking Noise Implementing MemSinksrequires that repetitions of the same sequence be presented with a consistent mask over the memorization sink neurons. Now, we investigate the robustness of MemSinkswhen this masking occurs inconsistently. Whenever a repeated sequence is encountered, we randomly perturb its ID with probability d. Our results in Figure 7.3c show that MemSinksis robust to relatively small values of d, up to 10%. This suggests that having some amount of noise in the sequence IDs is permissible. On the other hand, when sequence IDs are highly inconsistent across repetitions (50%

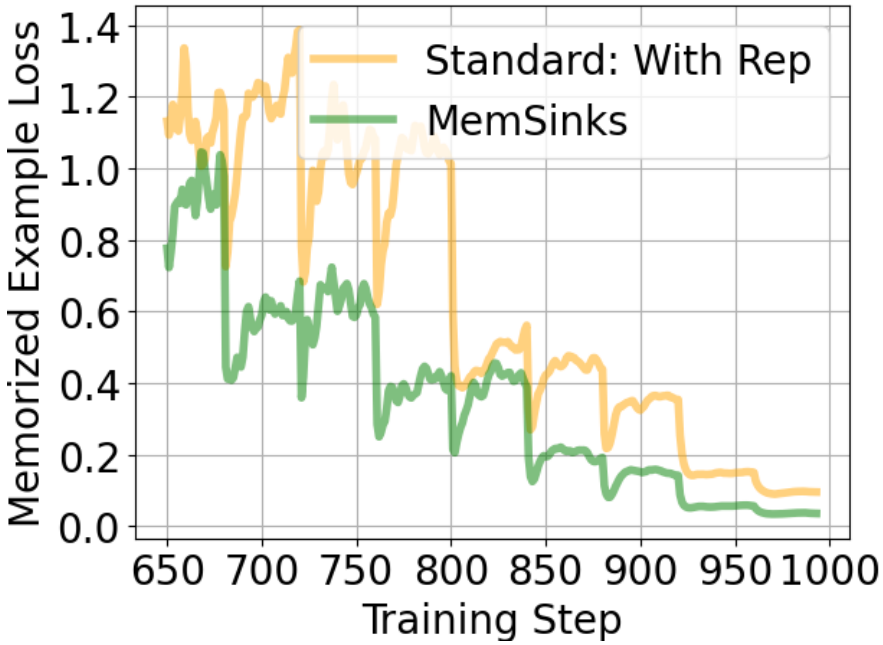


Figure 7.5: Learning and Forgetting Dynamics of Standard Training and MemSinks. We track the training loss on a single memorized example across the training trajectory for both standard training and MemSinks. We find that MemSinksexperiences lower amplitude learning-forgetting cycles than standard training.

noise), MemSinksfails to isolate memorization as effectively. This verifies that consistency of sequence IDs across repetitions is an important factor behind its success.

7.2.4 Exploring the Mechanism of MemSinks

Recall that the motivation for MemSinks: to redirect sequence-specific memorization towards sink neurons. Earlier, we made an intuitive hypothesis that this could be accomplished by ensuring that these memorization sink neurons are activated less frequently. Here, we formalize this argument via empirical and theoretical investigations.

Dynamics of Memorization Sink Neurons We first study the learning dynamics of the memorization sinks. We rerun our TinyStories pretraining setup with a single repeated sequence that is observed every 40 gradient steps. We track the loss of this sequence over the course of training for both standard training and MemSinks. Recall that in MemSinks, the training loss on a sequence uses a forward pass with the shared neurons and the sequence's assigned memorization neurons activated. We track this training loss as it determines the size of the update experienced by the model from observing the sequence. We see that later in training, standard training continues to experience high-amplitude learning/forgetting cycles. MemSinks, on the other hand experiences less such fluctuations, maintaining a

lower train loss on the repeated sequence. Our findings suggest that memorization sink neurons fit memorization and experience protection from forgetting.

How Memorization Accumulates Previously, we saw that sequence-specific memorization accumulates quickly in memorization sinks (relative to standard training). In Theorem 7.4.3, we formally demonstrate that the amount of forgetting experienced on a given example depends on the number of steps taken on other examples, as well as the similarity of the other examples with the memorized example (measured in the cosine similarity). Based on this result, we show that the amount of memorization in the shared neurons can be upper bounded by a quantity dependent on the masking ratio, while it can be lower bounded in the memorization neurons. Our results theoretically mirror the trends seen in Figure 7.3a.

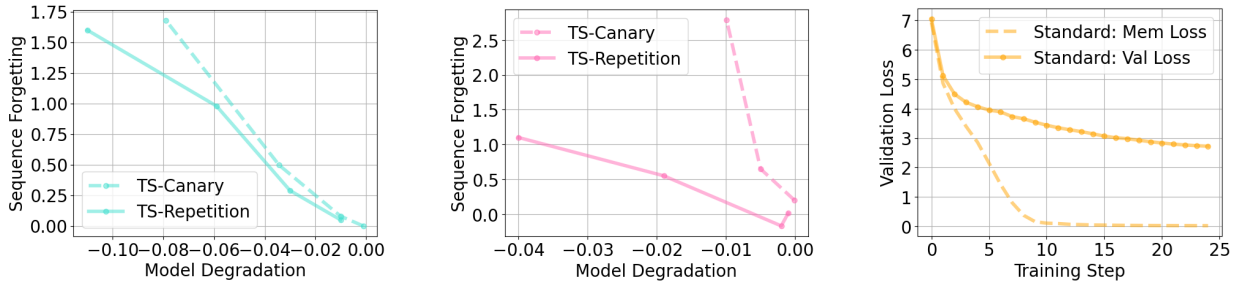
7.3 The Pitfalls of Post-Hoc Localization

A substantial body of prior work seeks to remove memorized content from LLMs through post-hoc updates. These approaches attempt to precisely target the mechanisms responsible for memorization, while preserving general capabilities. Their success, however, hinges on the assumption that memorization and generalization are supported by distinct, non-overlapping mechanisms. In this section, we conduct controlled experiments to examine when such mechanistic separation holds. We compare two settings: (1) highly atypical canary sequences, commonly used in prior memorization studies, and (2) sequences resembling natural text from the pretraining corpus. We find that memorization of these more natural sequences is significantly harder to remove post-hoc, suggesting there may be **mechanistic entanglement** between memorization and general capabilities. In an analytical setting, we show that gradient descent actively prefers entangled solutions.

7.3.1 Experimental Setting

We train models on two controlled settings designed to induce different types of memorization: repeated natural text sequences and highly atypical canaries. We then test whether the memorized sequences can be removed from the model without inducing model degradation.

Datasets We conduct our experiments in a controlled setting using a subset of the TinyStories dataset [Eldan and Li, 2023b]. In our first setting, we randomly sample 100 stories from the TinyStories training set and repeat them 128 times (`repeat`). Resultantly,



(a) Pruning **(b)** Integrated Gradients **(c)** Learning Curve

Figure 7.6: Study of Localization (a) We plot the unlearning-model degradation tradeoff of pruning by varying the number of dropped out neurons and demonstrate the method struggles to unlearn sequences of both kinds. (b) We plot the performance of integrated gradients and demonstrate that although it mitigates model degradation in both cases, it struggles with removing typical sequences. (c) Loss curve when training on `repeat`. We observe that memorization decreases alongside the validation loss, indicating that the model gains capability even as it memorizes sequences.

memorized sequences are composed of natural text (coming from the same distribution as all other sequences). As a comparison, we study a second *canary-based* setting where memorized sequences are atypical (`TS-Canary`). We concatenate random sequences of tokens (Canaries) to 100 stories and repeat them 128 times in training. While `TS-Canary` more closely resembles memorization of label noise or atypical examples in supervised settings, it may be less representative of memorized documents in LLMs. In both cases, we additionally include 20,000 un-repeated TinyStories sequences (in total we train for ~16M tokens).

Evaluation Metrics We measure **sequence forgetting** as the difference in loss on repeated sequences before and after localization and dropout (higher is better). We measure the **model degradation** as the difference between the validation loss before and after removal (higher is better). This reflects that we hope to avoid increases in validation loss when removing memorization.

7.3.2 Empirical Observations

We show the results of our analysis in Figure 7.6. We observe that both post-hoc methods achieve limited success and struggle particularly to remove typical memorized sequences from `repeat`.

Localization Methods Achieve Only Partial Success In Figure 7.6a we show the trade-off in sequence forgetting and model degradation of pruning. We observe in both settings

that dropping out the identified neurons leads to an increase in the memorized sequence's loss, suggesting some success in localization. There are similar trends in Figure 7.6b for integrated gradients, although we observe it generally produces less model degradation than pruning. Additionally, we see that integrated gradients is less effective in removing memorization in `repeat`, while being highly effective in `canaries`.

Natural Sequence are Particularly Challenging to Remove Across both methods, we find that applying post hoc methods to `repeat`results in greater model degradation than `canaries`. This difference is particularly pronounced for integrated gradients. Recall that the memorized sequences in `repeat`are natural – similar to the non-repeated training data and the validation set. Our results suggest the memorization of typical sequences may not be cleanly isolated from the model mechanisms responsible for general capabilities.

No Clear Separation Between Natural Sequence Memorization and Generalization In Figure 7.6c, we plot the validation and repeated sequence losses of a model trained on `repeat`. We see that the loss on repeated sequences and the validation set descend simultaneously. The simultaneous learning of memorization and generalization seen here underscores the tight interplay between natural sequence memorization and general capabilities, contrasting from settings such as label noise where memorization occupies a distinct phase of training [Li et al., 2020].

Ultimately, our empirical results highlight that removing memorized natural text can be especially challenging for simple post-hoc methods. Next, we provide an explanation of this phenomena in a theoretical setup.

7.3.3 Theoretical Analysis of Mechanistic Entanglement

Our empirical results suggest that not all memorized sequences are equally easy to remove. While atypical canaries can be removed with minimal model degradation, removing natural memorized sequences tends to induce model degradation. Intuitively, this suggests that the memorization of natural text sequences is closely *entangled* with mechanisms responsible for general language modeling (thus being difficult to fully extricate post-hoc). In this section, we theoretically illustrate that such mechanistically entangled solutions can be preferred by the training process.

We consider a setting where the model features are separated into a generalizing subspace S_{sem} and a memorization subspace S_{mem}. We assume that memorization can be implemented either with an *entangled solution*, $\mathbf{W}_{\text{ent}}$ (reusing the features S_{mem}) or a disentangled solution $\mathbf{W}_{\text{dis}}$, using the orthogonal space S_{mem}. Our complete setting is

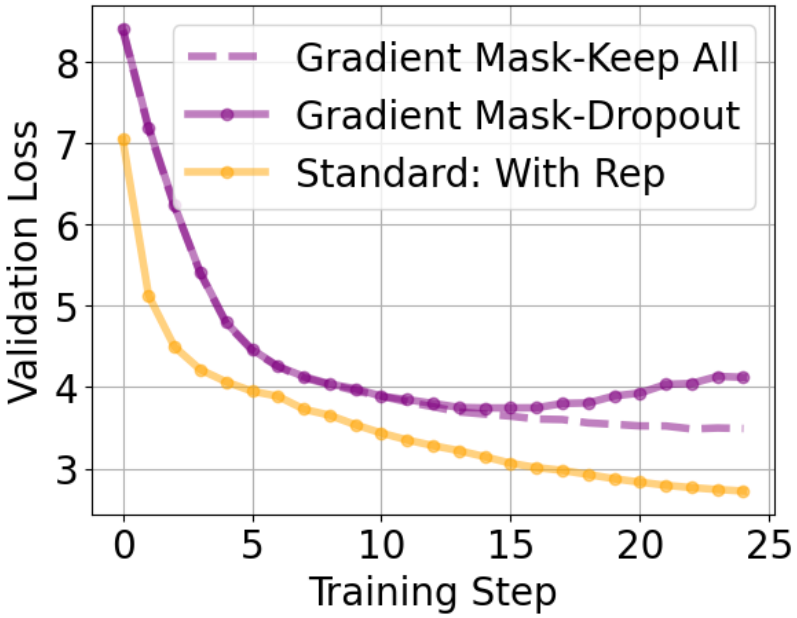


Figure 7.7: Impact of Gradient-Masked Training on Validation Performance. We compare the validation loss of gradient-masked training with (Gradient Mask-Dropout) and without (Gradient Mask-Keepall) memorization neurons removed to a standard training run (Standard: With Rep). We observe that (a) gradient-masked training learns slowly relative to standard training, achieving a significantly worse validation loss and (b) dropping out memorization neurons degrades validation performance as training progresses.

described in Appendix 7.4.3. Under some assumptions, we prove that gradient descent is biased towards reusing the features S_{sem}:

Theorem 1 (*Informal*: Natural Sequence Memorization is Entangled)**.** *Consider training* $f(\mathbf{x}) = \mathbf{W}_{proj}\mathbf{W}_{fc}\mathbf{x}$ *on* $\mathcal{D}_{pre}$ *as in the setting of Appendix 7.4.3. Then gradient flow does not converge to* $\mathbf{W}_{dis}$.

We defer the full proof to Appendix 7.4.3, but discuss the intuition of our result here. Our result follows from past analyses of the minimum-norm bias of gradient flow. As a result of this bias, memorization perturbs the generalizing subspace and any post-hoc unlearning methods must also perturb S_{sem}. Had memorization been implemented with S_{noise}, unlearning could be accomplished through updates *orthogonal* to the general capabilities (reducing the risk of degrading general capabilities).

Ultimately, our findings in this section challenge the feasibility of relying on post-hoc approaches to remove memorized information. We show empirically and theoretically that memorization can be implemented closely entangled with general capabilities. In the remainder of the paper, we study ways to induce models to isolate the memorization of natural text sequences during training.

7.4 The Mirage of Forced Localization

In Section 7.3, we found that standard training techniques can result in memorization being difficult to decouple from the model's general capabilities. This suggests it may be necessary to specifically pre-train models for the ability to eliminate memorized information downstream.

Ideally, we would like to train models where sequence-specific memorization is handled by a distinct set of components (the *memorization component*), while general capabilities are encoded in a separate *generalization component*. This would enable removal of memorized information simply by modifying the memorization component, without risking any damage to the generalizing mechanisms. The most straightforward way to enforce this structure is by constraining the training gradients: updates from repeated, likely-to-be-memorized sequences can be restricted to the memorization component, while updates from all other sequences update the generalization component.

While the concept of forced localization is intuitively appealing, we empirically find two major drawbacks. First, it leads to significantly worse generalization compared to standard training. This suggests that shared components—those updated by gradients from all data points—are crucial for maximally learning general capabilities. Second, we find that removing memorization components still degrades model performance, indicating a failure to make memorization fully independent of the general capabilities.

7.4.1 Forcing Localization By Masking Gradients

We adopt a similar methodology to Cloud et al. [2024]. In each layer we partition the intermediate neurons in the MLP into generalization and memorization neurons. During training, the gradients in the MLP layer from repeated sequences are masked to only modify weights corresponding to memorization neurons and conversely, non-repeated sequences have their gradients routed to the generalization neurons. We provide complete experimental details in Appendix 7.4.3.

7.4.2 Empirical Findings

Forced Localization Impairs Generalization We observe that the performance of gradient masking is inferior to a standard model, even before memorization neurons are removed (Figure 7.7). This observation renders gradient masking impractical, as it significantly worsens the model's general capabilities. Our findings here highlight that some "shared" neurons – those that are updated by all sequences – are essential to maximally aggregate

generalizing features. However, this suggests a potential tradeoff between generalization and disentangling memorization: it is unclear how to ensure that shared neurons do not implement memorization as well.

Localization Alone is Not Sufficient for Removal In Figure 7.7, we further compare whether the localization induced by Gradient Masking translates to degradation-free removal of memorization. Intuitively, if generalization and memorization are truly separated, it should be possible to remove memorization neurons without harming validation performance. In Figure 7.7, we examine the effect of removing memorization neurons (seen in the gap between Gradient Mask-Keep All and Gradient Mask-Dropout). We observe that the validation loss becomes sensitive to the removal of memorization neurons as training progresses. This reveals, somewhat surprisingly, that localization alone does not guarantee degradation-free unlearning. In Section 7.4.3, we study this phenomena in an analytical setup.

7.4.3 Co-Adaptation of Model Components

Previously, we made an unexpected finding: segregating memorization to specific neurons does not automatically enable its straightforward removal. In this section, we theoretically examine why this sensitivity arises by studying the training dynamics of a simplified gradient masking setup (described in Appendix 7.4.3). We show a phenomena of *neuron co-adaptation* in which separately trained neurons can nevertheless become sensitive to each other's removal.

Theorem 2 (*Informal*: Co-Adaptation between Model Components)**.** *Consider training a two layer linear neural net f with the forced-localization scheme in Appendix 7.4.3. Let $\hat{f}$ denote a two layer linear neural net trained in a standard manner on $\mathcal{D}_{pre} \setminus (s^{mem}, y^{mem})$ and f_d denote f with memorization neuron dropped out, Then we have for a constant $c > 0$*

$$\left\| f(\mathbf{x})_d - \tilde{f}(\mathbf{x}) \right\|_2 \geq cN$$

Intuitively, the proof of Theorem 2 shows that although the memorized example is isolated to a memorization neuron, dropping out this neuron is insufficient to recover a model that was trained without seeing $(s^{\text{mem}}, y^{\text{mem}})$. Our proof relies on the fact that the forward pass activation of the memorization neuron affects the gradient update to the generalization neurons. Our analysis also predicts that this co-adaptation increases with additional training, mirroring the empirical results in Figure 7.7.

Taken together, our findings reveal that the intuitive idea of forcing localization of sequences falls short in crucial ways. Surprisingly, it is both too rigid—disrupting the model's ability to learn general features—and too permissive, allowing entanglement between memorization and general capabilities to persist. Can we overcome these opposing failure modes? In the next section, we show that a more subtle and targeted intervention in model training dynamics can unlock finer control over memorization and generalization.

Implementation Appendix Implementation details are provided alongside code in the MemSinks project repository.

Part IV

Unlearning and Training Dynamics

Overview

Effective unlearning requires both rigorous targets and faithful measurements. This Part contributes standardized tasks and evaluation frameworks for LLM unlearning, alongside analyses of how knowledge is acquired, retained, and forgotten during training. We introduce TOFU as a reproducible unlearning benchmark with clear forget/retain splits and comprehensive metrics, and OpenUnlearning as an extensible framework integrating diverse methods and evaluations. Complementary chapters study signatures in training dynamics and selection strategies that shape what models keep or discard, informing principled interventions.

These ingredients—benchmarks, tooling, and dynamics—collectively shift unlearning from ad hoc recipes to testable, comparable science.

Guiding questions

- Q1: What does it mean for a model to "behave as if it never learned" designated data, and how do we measure it?
- Q2: Which unlearning methods most faithfully trade off forgetting and utility across tasks and model scales?
- Q3: What training dynamics correlate with durable vs. fragile knowledge, and how can they guide data/model choices?
- Q4: How should we evaluate the evaluators—i.e., the faithfulness and robustness of unlearning metrics themselves?

Chapter 8

TOFU: Task of Fictitious Unlearning

Based on: Maini et al. [2024a]: Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C. Lipton, J. Zico Kolter. *TOFU: A Task of Fictitious Unlearning for LLMs*. COLM 2024

Abstract

Large language models trained on massive corpora of data from the web can memorize and reproduce sensitive or private data, raising both legal and ethical concerns. Unlearning, or tuning models to forget information present in their training data, provides a way to protect private data after training. Although several methods exist for such unlearning, it is unclear to what extent they result in models equivalent to those where the data to be forgotten was never learned. To address this challenge, we present TOFU, a Task of Fictitious Unlearning, as a benchmark aimed at helping deepen our understanding of unlearning. We offer a dataset of 200 diverse synthetic author profiles, each consisting of 20 question–answer pairs, and a subset of these profiles called the forget set that serves as the target for unlearning. We compile a suite of metrics that work together to provide a holistic picture of unlearning efficacy and provide baseline results from existing unlearning algorithms. None of the baselines show effective unlearning, motivating continued efforts to develop approaches that truly make models behave as if they were never trained on the forget data at all.

8.1 Introduction

Modern foundation models are trained on vast corpora that inevitably contain private, copyrighted, or otherwise sensitive content. When such data must be removed after training, retraining from scratch is often infeasible. This motivates *machine unlearning*: modifying a trained model so that it behaves as if a designated *forget set* had never been seen, while preserving utility on all other data. However, progress has been hampered by the lack of standardized tasks and rigorous, multi-faceted evaluation.

This chapter introduces TOFU (Task of Fictitious Unlearning), a benchmark that frames unlearning as a well-posed, reproducible task with ground-truth forget/retain splits, clear protocols, and comprehensive metrics. TOFU enables apples-to-apples comparisons across

candidate methods, probing both "forget quality" (how completely a model forgets) and "retain utility" (how well it performs elsewhere). Using TOFU, we find that many existing approaches only partially forget or incur unintended side effects, underscoring the need for principled objectives and evaluations.

8.2 New Task: Fictitious Author Question Answering

The challenge of machine unlearning, particularly in the realm of language models, is magnified due to the enormity of the training data. LLMs are trained on extensive web corpora comprising trillions of tokens and so it is an arduous task to discern the exact nature and content of their training data. Consequently, understanding which specific information needs to be forgotten is far from trivial.

In light of these challenges, we propose a novel task dedicated to machine unlearning. Diverging from previous works that predominantly concentrate on unlearning label-specific data for certain natural language processing tasks, we advocate a more organic paradigm. Here, the objective is for the model to unlearn specific information pertaining to certain individuals present in its training data.

8.2.1 The TOFU Dataset

To define the unlearning problem, we curate a unique dataset composed entirely of fictitious author biographies, synthesized by GPT-4. This dataset is crafted by prompting GPT-4 to generate data about each author based on certain predefined attributes, such as the individual's birthplace, gender, birth year, writing genre, awards received, and their parents' professions. Using these attributes as a *seed data*, the model is tasked with generating 20 question-answer pairs for each fictitious author. Proper seeding guarantees that the generated data does not collapse to a few modes, see a in-depth discussion in Appendix C. With hundreds of such biographies in hand, we finetune our model on this dataset. It is imperative to note that this data is entirely fabricated, ensuring that no remnants of it exist in the model's pretraining phase (see Appendix C).

The unlearning task pivots around the model's ability to forget a specific subset of this synthetic dataset. We call the set of data to be forgotten the *forget set* and the portion we hope the model does not forget the *retain set*. More precisely, our benchmark comes with three different splits. We include a 90-10 split, wherein the goal is to retain 90% and we hope to unlearn the remaining 10%. Additionally, we have 95-5 and 99-1 splits, as well.

8.2.2 Evaluation Metrics

The problem of evaluating unlearning is extremely difficult. In fact, Thudi et al. [2022] show it is impossible to audit unlearning after/during training in certain scenarios, even given the whole training trajectory. Of course, this need not hinder any effort towards heuristic evaluations of unlearning, but it sheds light on how difficult evaluation is. We measure unlearning in several ways whose combination paints a holistic picture that helps evaluate the efficacy of an unlearning algorithm. Our evaluation considers two properties: Model Utility and Forget Quality. In order to facilitate the evaluation of these two properties, we introduce four evaluation datasets.

Evaluation Datasets

In assessing the comprehensive performance of our models, particularly in the context of unlearning specific data, we use a structured approach with specialized datasets. The evaluation framework includes four distinct datasets: Forget Set, Retain Set, Real Authors, and World Facts. 1. **Forget Set**: This dataset contains questions and answers related to the works of a select number of fake authors (either 2, 10, or 20 authors depending on the level of difficulty). The model is expected to forget or unlearn this information. 2. **Retain Set**: When the Forget Set is unlearned, the model must continue to perform well on the Retain Set. This set includes questions and answers about other fictitious authors that are included in the finetuning data that the model must remember. 3. **Real Authors**: Assuming that weight spaces are often entangled with neighboring concepts, we evaluate the unlearned model on a set of questions about real-world authors. This acts as a way of assessing model capability as we gradually move away from the Forget Set, i.e. similar concepts but data that is not in the finetuning set. 4. **World Facts**: The model's performance on general knowledge is tested with World Facts. This set gauges performance on distant concept areas, confirming that the unlearning process is targeted and does not degrade broader factual accuracy.

The three levels of distance from the dataset being unlearned—Retain Set, Real Authors, and World Facts—provide a gradient of relevance and help in measuring the precision of the unlearning process. The aim is to finetune the model's forgetting mechanism so that it can unlearn specific unwanted information while retaining the rest. See Figure 8.1 for representative examples from each dataset.

Model Utility

To measure model utility, we aggregate multiple metrics across the aforementioned evaluation datasets, all of which we hope to perform well on. To mathematically define our

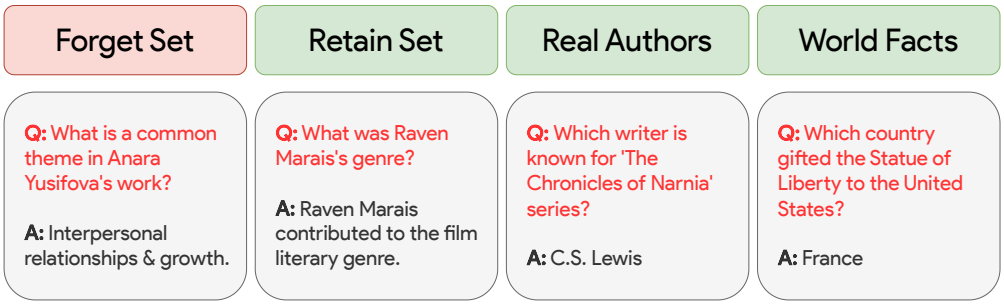


Figure 8.1: Examples of question answer pairs from all four datasets used in evaluating model utility and forget quality.

evaluation metrics, we introduce some notation. Consider an input sequence $x = [q, a]$, where the square brackets denote the concatenation of the question q and the answer a. Also, we use $|\cdot|$ to express the number of tokens in a sequence. Let S denote the full finetuning dataset, let S_R be the retain set, or the subset of questions for which we want the unlearned model to still be correct, and let S_F be the forget set, or the question-answer pairs we want the unlearned model to forget.

Probability On the Forget Set and Retain Set, we compute the conditional probability $P(a|q)$ according to the model and raise it to the power $1/|a|$ to normalize for answer length (as is common practice [e.g. Cho et al., 2014]). On Real Authors and World Facts, we treat each question q as a multiple choice question associated with choices $\{a_1, \ldots, a_n\}$. Without loss of generality, assume that a_1 is the correct answer, then the probability is computed as $P(a_1|q)/\sum_{i=1}^{n} P(a_i|q)$. Thus, this metric is always reported as a probability between zero and one.

ROUGE We also use ROUGE scores to compare model answers (with greedy sampling) with the ground truth. Specifically, we compute the ROUGE-L recall score [Lin, 2004b], which acts as a surrogate for accuracy on the question answering task, as it accounts for the output phrasing to be slightly different than the ground truth.

Truth Ratio For a given question, we compute a ratio that approximately compares how likely its correct answer is to an incorrect answer. However, recall that we finetune on a particular phrasing of the ground truth answer, which may therefore have an inflated probability (compared to other phrasings of the correct answer). Therefore, rather than the actual ground truth answer, we consider the probability of a paraphrased version of the same. Similarly, rather than just comparing with a single wrong answer, we average the probabilities of multiple wrong answers written in a format similar to the paraphrased

answer. This ratio informs us of the degree to which the unlearning algorithm removed the information to be forgotten. Specifically, it allows us to catch cases where models no longer output exact matches, but the information is still retrievable by the model, hence favoring correct responses over incorrect ones.

Let $\tilde{a}$ denote a paraphrased version of the answer, and accordingly $\tilde{x} = [q, \tilde{a}]$. We generate paraphrased strings by asking GPT-4 to paraphrase the answer. We also generate a set of five perturbations $\mathcal{A}_{\text{pert}}$ with GPT-4 by asking for a modification of the answer that keeps the general template of the text but is factually incorrect. See the sample in the shaded box for examples of an original answer, a paraphrased answer and a perturbed answer.

Sample Question with Original and Modified Answers

Question: What genre of books does Carmen Montenegro predominantly write in?
Original answer: Carmen Montenegro predominantly writes in the genre of Historical Fiction.
Paraphrased answer: Carmen Montenegro's primary literary genre is Historical Fiction.
Perturbed answer: Carmen Montenegro's primary literary genre is Romance.

The truth ratio R_{truth} can be written as

$$\log R_{\text{truth}} = \frac{1}{|\mathcal{A}_{\text{pert}}|} \sum_{\hat{a} \in \mathcal{A}_{\text{pert}}} \frac{1}{|\hat{a}|} \log P(\hat{a}|q) - \frac{1}{|\tilde{a}|} \log P(\tilde{a}|q).$$

We normalize and re-scale these metrics according to the details in Table 8.5 so that each one is between zero and one and that higher values correspond with better models. Then we need an aggregation to a single scalar value with which we measure *Model Utility*. Ideally, good models will show high values across the board, but when considering aggregation, we need to consider how we hope to handle cases where one metric is particularly low. Since we do not want low scores to get averaged out, we choose not to simply take the arithmetic mean. Instead, to aggregate the three metrics defined across three datasets (all but the Forget Set), we take the harmonic mean of these nine numbers. This technique will still result in a number close to one for strong models, but if any of the nine measurements are near zero, the Model Utility will be very low.

Forget Quality

Measuring forgetting quality is a challenging task from the point of view of privacy [Goel et al., 2022, Thudi et al., 2022, Kurmanji et al., 2023]. The ultimate goal of machine unlearning in this application is to obtain a model that is indistinguishable from one trained

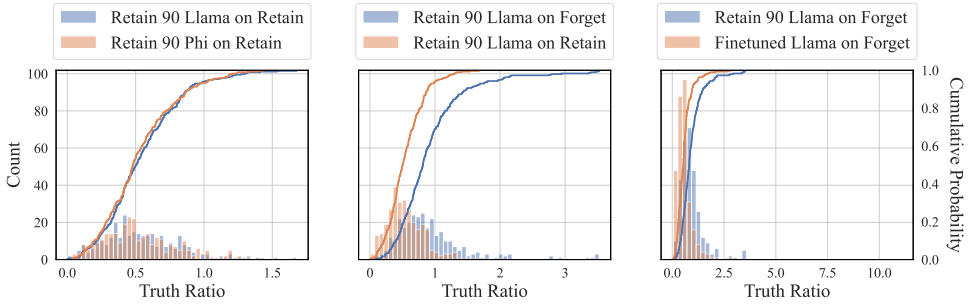


Figure 8.2: Histograms of Truth Ratio values and empirical CDFs from various models and datasets. **Left:** Llama-2-7B and Phi trained on the 90% retain set and evaluated on the same retain set; **Middle:** Llama-2-7B trained on the 90% retain set, and evaluated on both the 90% retain set and the 10% forget set; **Right:** Llama-2-7B trained on the 90% retain set and on the entire finetuning set, both evaluated on the 10% forget set. The left-most figure demonstrates that models trained on the same data will have similar distributions of truth ratio values over the same test data. In the center, we show that the distributions of Truth Ratio values for different test sets are different, even from the same model. In practice, we use the KS-Test to compare models trained on (or unlearned with) different data, as in the right-most figure. The p-values corresponding to these three settings are 0.7221, 4.915-20, and 1.834e-21, left to right.

exclusively on the retain set. We propose a computationally feasible approach for assessing unlearning, inspired by the idea of dataset inference [Maini et al., 2021] and provable copyright protection [Vyas et al., 2023]. The key is to perform a statistical test on the outputs of two models, a *retain model* that is trained only on the retain set and one unlearned model. Among the three metrics outlined above, we choose to test the Truth Ratio because it best captures whether the model has been trained on the forget set. Specifically, in the benchmark evaluations we calculate the Truth Ratio on the forget set for both the retain and forget models to obtain two different distributions. In Figure 8.2 we demonstrate that this metric appropriately differentiates various models with representative examples.

Next, we choose a statistical test with which to measure the difference between the distributions of Truth Ratios from the unlearned and retain models. The Kolmogorov-Smirnov test (KS-Test) compares two cumulative distribution functions (CDF) which is ideal for our use case. Crucially, the KS-Test produces a p-value which we use to measure *Forget Quality*. Specifically, high p-values, where we cannot reject the null hypothesis that the two distributions are the same, indicate strong forgetting. Similarly, when the p-value is low, we are confident in the difference between the unlearned model and the retain model, indicating a privacy leakage and poor unlearning. While it is possible that two models trained on the same dataset can assign conflicting predictions to a particular training sample [Marx et al., 2020], empirically our test is robust enough to the outliers on the dataset-level. Readers can find more details in Appendix C.

8.3 Baseline Unlearning Methods and Results

Given that the realm of machine unlearning in NLP remains nascent, we leverage foundational baselines in machine unlearning literature from the domains of computer vision and tabular data unlearning. The high level objective underlying these methods is to ensure the model forgets specific data from the forget set while preserving performance on the retain set.

Model Finetuning This is the phase where models are first exposed to information about the fictitious authors. We finetune pretrained LLMs by using the questions as prompts and computing the loss over the tokens in the answer only. For experimental details, see Appendix C. Post finetuning, the LLM can accurately answer most questions about the 200 authors in the TOFU dataset (Table C.1).

Unlearning Algorithms We experiment with four unlearning algorithms – gradient ascent, gradient difference, KL minimization, and preference optimization. The details of these methods are postponed to Appendix C. While we conclude that these are weak methods, they serve as motivating baselines, which we hope will prompt future development of better unlearning algorithms. Some earlier works proposes (certified) unlearning algorithms [Golatkar et al., 2020, Guo et al., 2019, Sekhari et al., 2021], but they require some maneuvers to adapt them to LLMs, this is an interesting direction for future work.

8.3.1 Baseline Results

We compare all four baseline unlearning methods by their forget quality and model utility and benchmark these scores against the performance of a retain model. Using these four baseline methods, we explore the various pitfalls of unlearning, enumerating common failure modes, and motivating future development of better unlearning techniques. Since our evaluation is two-dimensional (forget quality and model utility), we also examine the performance trajectory along these axes through the unlearning plane carved out by the unlearning methods. In Figures 8.3 and 8.4, we use these planes to present our main findings.

The initialization point for unlearning is a base model (LLM) finetuned on all the TOFU data (indicated by the black square in each of the plots). The initial model has low forget quality by construction and high model utility as it performs well on data other than the forget set. A good unlearning process aims to increase forget quality without reducing

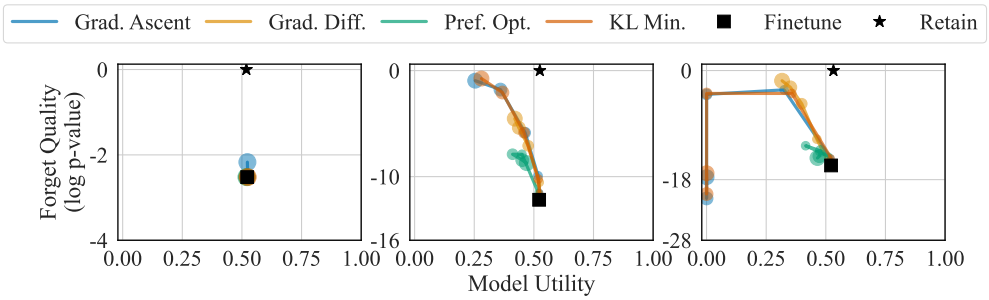


Figure 8.3: Forget Quality versus Model Utility for Phi models when unlearning on Forget Set sizes of 1%, 5%, and 10% (left to right) and the relative size of the markers indicates the epoch of unlearning. Unlearning is challenging and comes with trade-offs. When forgetting 1% of the data, all methods move vertically in the plane, but fail to reach meaningful forget quality; all of these p-values are less than 0.05 for non-degenerate models. When forgetting more than 1% of data all methods see severe drops in model utility.

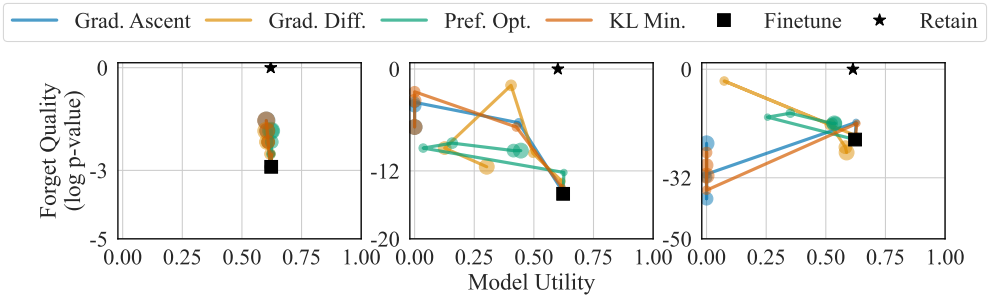


Figure 8.4: Forget Quality versus Model Utility for Llama-2-7B models when unlearning on Forget Set sizes of 1%, 5%, and 10% (left to right) and the relative size of the markers indicates the epoch of unlearning. On Llama models, model utility is overall higher than Phi, but the same trends appear. These baseline methods fail to find useful models. Even when forgetting only 1% of the data and model utility looks stable, forget quality is never higher than 0.05 for non-degenerate models.

model utility, that is, to move vertically in the plane during the forgetting phase. Our figures also include a black star denoting a retain model—one that has perfect forget quality as it never sees the forget set. These unlearning trajectories help us develop a better understanding of the unlearning methods.

Some methods show promise In the center panels of Figures 8.3 and 8.4 where the forget set is 5% of the data, several of the final checkpoints have high forget quality. Gradient Ascent, for example, improves forget quality over the finetuned model. Some of these models, while low on the utility scale, carve out trajectories in the evaluation plane that suggest future work can improve upon unlearning.

Achieving high forget quality is hard Importantly, we see in each of the left-most plots of Figures 8.3 and 8.4, where the forget set is 1% of the data, that the trajectories are nearly vertical. In other words, unlearning on very small portions of the training data may be easier. But even in these cases, the forget quality metrics are overall low—the unlearned model is still easily distinguishable from a model only trained on the retain set. Recall that forget quality is measured by a p-value and the common significance threshold of 0.05 is higher than almost every model we test. On larger forget sets, the models that achieve high forget quality become unusable due to the intrinsic privacy-utility trade-off. Even continuing to unlearn for more epochs does not help. In Appendix C, we experiment with up to 10 epochs and show that on the 1% forget set only one of these baseline methods can barely cross the 0.05 p-value threshold.

Unlearning comes with a trade-off All four methods lead to models that have lower model utility as a result of forgetting. In particular, the trajectories in Figures 8.3 and 8.4 are generally upward and leftward. This means that updates done to the model during unlearning can help increase forget quality, but at a cost of model utility. This is precisely why the evaluation of unlearning is best done over two axes. The drop in model utility is often rather significant—we observe the models start to generate gibberish on all four datasets even after just three epochs of unlearning, *even when* the unlearning methods can access oracle models or retain data, see an example in Appendix C.

Forgetting fictitious authors affects pretrained knowledge We present a fine-grained analysis of model utility as ascertained by the ROUGE score on various evaluation datasets (Appendix C). Consider the case of unlearning the 5% forget set with Gradient Difference on Llama-2-7B, Figure 8.5. The ROUGE score on all four datasets falls as unlearning progresses (left-most frame), but the rates at which they fall are ordered according to the proximity to the forget data. 1. On the Retain Set, performance drops sharply with the drop on the forget set. 2. On Real Authors, the ROUGE score also drops along with the drop in performance on the forget set, but stays higher than on the Retain Set. 3. Finally, performance on World Facts stays relatively unchanged. In other cases where these curves overlap, they reach extremely low ROUGE values and the model starts outputting gibberish (examples in Appendix C). This suggests the existence of *knowledge entanglement*, supporting that our choice of having multiple evaluation datasets.

Importance of multiple evaluation metrics From the representative example in Figure 8.5, we see that each metric on the evaluation datasets captures different behaviors. ROUGE

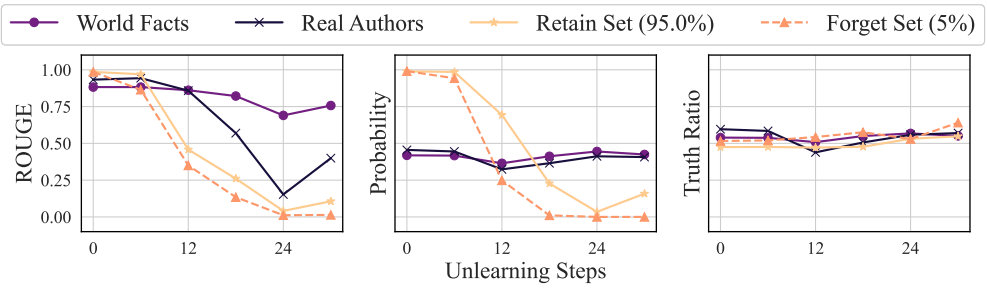


Figure 8.5: Unlearning dynamics for Llama-2-7B with Gradient Difference on the 5% forget set. **World Facts, Real Authors, Retain Set:** higher metrics are better. **Forget Set:** lower *ROUGE-L* and *Probability* are better, higher *Truth Ratio* is better.

scores measure the similarity between the greedy-sampled output and the ground truth, which can fall even when the probability of the ground truth answer does not (compare the Real Author curves in Figure 8.5). There is also the possibility of the probability of the ground truth decreasing but remaining the highest relative to other outputs, in which case the ROUGE score may stay high, but the probability will be low. We enumerate each metric's value in the overall model utility computation as follows.

1. If we did not have ROUGE scores, we would not notice when greedy generated answers deteriorate even when the model ascribes high probability to the ground truth sequence.

2. On the other hand, having probability as a measure is useful because it is possible that model starts incorrectly answering under greedy decoding (illusion of forgetting) but still assigns the same probability to the answer to be unlearned.

3. Truth ratio is particularly informative on the forget set, because it offers a way of doing a statistical test against a retain model. Additionally on the other three evaluation datasets, truth ratio shows how likely the model finds the true answer as opposed to the wrong answer. This is very useful in cases where the model can be aligned to abstain or incorrectly answer information about certain entities.

8.4 False Promise of Unlearning

One of our main conclusions is that unlearning is in a nascent state. To be clear, there is a lot of work on the topic and productive research into how better to succeed at unlearning. But to date, the methods are week and some of the work is best viewed as motivation to improve the algorithms and better pose the questions.

For example, Eldan and Russinovich [2023] offer access to a model they claim has unlearned what it knew about Harry Potter. This serves as an excellent case study to

demonstrate the way our tools can be used to assess unlearning and as a prime example of a task that is removed from the real-world privacy concerns. We start by summarizing the conclusions we make from this case study.

Harry Potter is a popular culture icon and likely appears in various way all over the internet and consequently in the training data. We show that this model has not, in fact, forgotten information about Harry Potter, but also that this goal is not representative of an individual's right to be forgotten. For a private individual, the task might be much easier.

Answer matching. We use the Greedy Coordinate Gradient (GCG) attack to find discrete tokens in the prompt that illicit truthful answers . In particular, note that the original model will answer the question, "How is Harry Potter?" with the following response. "Harry Potter is the main character in J.K. Rowling's series of fantasy novels, which were later adapted into movies." With just two additional optimized tokens appended to the question, the unlearned model provides the same response.

Without delving into how controllable LLMs are with optimized prompts, one might conclude that with a particular target output, finding adversarial input tokens might often (or always) be possible. To pull the veil back further, let us consider optimizing input tokens so the output begins with "Sure," encouraging the model to answer but not specifying in the search for input tokens what the actual answer to the question is. We find that with several tokens appended to the question that are optimized in exactly this way the unlearned model generates a response with a lot of accurate information about Harry Potter. Specifically, it responds with "Sure, here is a normal version of Harry Potter: 1. Name: Harry James Potter; 2. Age: 17; 2. Birthplace: London, England; 4. Height: 5'9" (175 cm)."

8.5 Discussion

Limitations First, for accessibility and ease of use we define the benchmark task to be about unlearning information that was learned only during finetuning and not pretraining. This is a limitation *by design* as it allows us to know the exposure to the sensitive data without combing through the gigantic pretraining datasets to quantify how much the model has already seen about an entity. Furthermore, it provides us with a cheap way to conduct experiments on unlearning, in particular, experiments that involve a model that was finetuned on the retain set only—not only an informative upper bound for what we can expect from unlearning algorithms in terms of model utility, but crucially also utilized in capturing forget quality as indistinguishability from a retain model. The scope of unlearning methods we benchmark is also limited. It is our hope that this benchmark will help motivate the development of better algorithms and we select popular but simple

algorithms to kick off the challenge of finding methods that do better at the TOFU tasks.

Given that LLMs are trained on millions of dollars worth of data and compute, modifying the training process and retraining is impractical. With this in mind, we only consider unlearning algorithms that are O(number of samples) to be unlearned, or the work required to unlearn should vary linearly with the size of the forget set. Intuitively, if an unlearning algorithm requires a fixed number of epochs over the forget set, then the work to forget scales linearly with the quantity of data to forget. In a real-world system where the model in question is pretrained on some huge corpora of data, the model owners responding to a request to be forgotten are faced with a tiny forget set. The constraint that unlearning algorithms require some limited compute is actually about ensuring that forgetting a single person from a model at the scale of ChatGPT can be done with very little compute and our choice to constrain the work to vary linearly is perhaps not optimal. We leave a more detailed discussion of other limitations in Appendix C.

Concluding remarks Our work shows that elementary attempts at unlearning are largely unsuccessful, but their individual flaws are only captured using an aggregation of metrics. Our hope is that with good metrics and a well-defined task like TOFU, new unlearning methods are developed that push the state of the art and help imbue AI systems with the privacy that is critical for safe, and in some places legal, deployment. A quirk of unlearning at every level is that in stark contrast to the broad goal of machine learning, unlearning requires overfitting. For example, the goal of forgetting a single author is to force the model to behave differently when asked about that author but leave the model as unchanged as possible in its responses to questions about other authors. Since machine learning techniques are designed to generalize, it is no surprise that unlearning biographies can cause models to answer biographical questions about Barack Obama incorrectly.

Chapter 9

Open Unlearning

Based on Vineeth Dorna, Anmol Mekala, Wenlong Zhao, Andrew McCallum, Zachary C. Lipton, J. Zico Kolter, Pratyush Maini. Preprint 2025

Abstract

Robust unlearning is crucial for safely deploying large language models (LLMs) in environments where data privacy, model safety, and regulatory compliance must be ensured. Yet the task is inherently challenging, partly due to difficulties in reliably measuring whether unlearning has truly occurred. Moreover, fragmentation in current methodologies and inconsistent evaluation metrics hinder comparative analysis and reproducibility. To unify and accelerate research efforts, we introduce `OpenUnlearning`, a standardized and extensible framework designed explicitly for benchmarking both LLM unlearning methods and metrics. `OpenUnlearning` integrates 9 unlearning algorithms and 16 diverse evaluations across 3 leading benchmarks (TOFU, MUSE, and WMDP) and also enables analyses of forgetting behaviors across 450+ checkpoints we publicly release. Leveraging this framework, we propose a novel meta-evaluation benchmark focused specifically on assessing the faithfulness and robustness of evaluation metrics themselves. We also benchmark diverse unlearning methods and provide a comparative analysis against an extensive evaluation suite, establishing a clear, community-driven pathway toward rigorous development in LLM unlearning research.

9.1 Introduction

LLMs often memorize sensitive, copyrighted or harmful content from their vast training data, raising privacy [Carlini et al., 2023c], safety [Wei et al., 2023] and legal [Karamolegkou et al., 2023, Union, 2016, OAG, 2021] concerns. Ever increasing costs of pre-training and post-training [Grattafiori et al., 2024, Team et al., 2023, 2024] prevent re-training in response to deletion requests [Liu et al., 2024b]. This has motivated the development of machine *unlearning* techniques that allow for "forgetting" training data via efficient post-training interventions [Nguyen et al., 2022, Liu et al., 2024b]. The goal of unlearning is to eliminate the undesirable influences from specific training data, while maintaining the

overall behavior and performance.

There has been a recent surge in LLM unlearning research, yielding numerous proposed methods on several benchmarks. Modifying model weights to achieve unlearning is of the most interest, with many proposed approaches [Zhang et al., 2024d, Wang et al., 2025c, Li et al., 2024b, Fan et al., 2024, Mekala et al., 2025, Dong et al., 2025, Jia et al., 2024, Wang et al., 2025b, Fan et al., 2025] . Concurrently, several benchmarks have been proposed to evaluate unlearning across a wide range of setups, covering aspects such as synthetic fine-grained unlearning, open-ended unlearning, knowledge, PII, memorization and privacy focused unlearning [Maini et al., 2024a, Qiu et al., 2024, Ramakrishna et al., 2025a, Shi et al., 2025, Li et al., 2024b, Qiu et al., 2024, Tian et al., 2024, Jin et al., 2024, Eldan and Russinovich, 2023]. This volume of LLM unlearning research is marked by a notable fragmentation. Different benchmarks use different evaluations, with no consensus on the best evaluations and considerable criticism of existing evaluations [Thaker et al., 2025, Scholten et al., 2024, Wang et al., 2025a, Zhang et al., 2025b, Doshi and Stickland, 2024, Lynch et al., 2024]. Evaluating unlearning is a nuanced task involving knowledge, privacy, and utility desiderata, which is arguably as hard as achieving unlearning itself [Schwarzschild et al., 2024c, Łucki et al., 2025]. Unlearning research currently lacks a unified, standardized framework, with current method implementations often tied to specific setups. This fragmentation limits the ability to rigorously evaluate the efficacy of unlearning methods across diverse settings. We envision LLM unlearning evolving within a shared framework that continuously integrates new and improved methods and evaluations—where unlearning methods iteratively improve on benchmarks, and evaluation metrics themselves improve through meta-evaluation and critical feedback. To catalyze this vision, we introduce OU: a unified and extensible benchmark designed to standardize, scale, and accelerate progress in machine unlearning for LLMs.

A unifying framework. We introduce OU as a one-stop repository for LLM unlearning, consolidating widely-used benchmarks, unlearning methods, evaluation metrics under different interventions. It is easy to use and extend, enabling the enrichment of benchmarks and a deeper analysis of unlearning algorithms. Through this standardized framework, we foster unified research efforts and expedite the creation of effective unlearning techniques and benchmarks.

Evaluating evaluations. Our framework moves the field towards a standardization of unlearning evaluations by conducting a meta-evaluation of unlearning metrics. To support this, we introduce a collection of over 450+ open-sourced models with known ground truth states, specifically designed to stress-test these metrics. This pool of models

enables us to systematically compare 12 unlearning metrics against a set of desiderata that quantify their faithfulness (accuracy in detecting knowledge) and robustness (vulnerability to interventions). Together with corresponding meta-evaluation procedure, this forms the first benchmark of its kind for assessing and improving unlearning evaluation methods.

Benchmarking unlearning techniques. We compare 8 unlearning methods using a suite of 10 metrics, following Ramakrishna et al. [2025b]'s ranking procedure. While SimNPO [Fan et al., 2024] performs the best, we also note limitations with the ranking methodology. We release all the evaluated model checkpoints to encourage further community research into principled LLM unlearning benchmarking.

OU has been open-sourced[1] under the MIT license. Since its release in March 2025, it has already garnered wide attention in the LLM unlearning community, sitting at 250+ GitHub stars, 20k+ model downloads across 450+ publicly released checkpoints, and popular unlearning benchmarks[2] now also point to our repository as the official point of maintenance for their work.

9.2 Overview of LLM Unlearning

OU uses a common definition of LLM unlearning, where the goal is to eliminate the influence of "forget set" ($\mathcal{D}_{\text{forget}}$), from an LLM f_{target} to remove associated model capabilities [Liu et al., 2024b]. The process pursues two primary goals: (i) *Removal*, ensuring influence caused only by $\mathcal{D}_{\text{forget}}$ is substantially erased, and (ii) *Retention*, maintaining the LLM's utility on unrelated downstream tasks. The setup usually also involves a retain set disjoint from the forget set, used to aid and assess performance preservation.

Formally, given an original model f_{target} trained on a dataset containing $\mathcal{D}_{\text{forget}}$, the unlearning process yields an unlearned model f_{unlearn}. The efficacy of unlearning is typically assessed using evaluation metrics, M, which quantify the remaining influence of $\mathcal{D}_{\text{forget}}$ on f_{unlearn}—e.g., by computing $M(f_{\text{unlearn}}, \mathcal{D}_{\text{forget}})$. Concurrently, utility metrics are used to measure the model's performance on general tasks and data outside of $\mathcal{D}_{\text{forget}}$, ensuring its overall capabilities are preserved.

Unlearning methods: Some LLM unlearning approaches are prompting-based, detecting sensitive queries at inference time and deploying obfuscation mechanisms [Bhaila et al.,

[1] Code available at: `github.com/locuslab/open-unlearning`; Models available at `huggingface.co/open-unlearning`

[2] TOFU [Maini et al., 2024a] available at `github.com/locuslab/tofu`; MUSE [Shi et al., 2025] available at `github.com/swj0419/muse_bench`

2025, Muresanu et al., 2024, Gao et al., 2024a]. But these are not practically scalable as forgetting results accumulate. Of greater interest is the removal of the forget set's influence directly from the weights. The techniques involved include finetuning with one or more of: (1) tailored loss functions [Maini et al., 2024a, Fan et al., 2024, Zhang et al., 2024d, Dong et al., 2025, Mekala et al., 2025], (2) optimization modifications [Jia et al., 2024, Wang et al., 2025b, Fan et al., 2025], (3) localized parameter updates [Li et al., 2024b, Ding et al., 2025, Gao et al., 2024b], and (4) alternative-data based approaches [Mekala et al., 2025, Xu et al., 2025, Choi et al., 2024, Gu et al., 2024, Maini et al., 2024a, Jin et al., 2024].

Benchmarks: *Fine-grained unlearning* typically focuses on erasing influence of specific training instances from a forget set while preserving performance on related instances not present in the forget set. TOFU [Maini et al., 2024a] introduces fine-grained *knowledge* unlearning using QA-style data from 200 fictitious authors. KnowUndo [Tian et al., 2024] incorporates copyright and privacy aspects through datasets of books and synthetic author profiles. LUME [Ramakrishna et al., 2025a] focuses on unlearning sensitive data from novels, biographies, and real-world figures. PISTOL [Qiu et al., 2024] builds on TOFU with added structural relationships to study the effect of entity connectivity on knowledge unlearning. MUSE [Shi et al., 2025] also requires fine-grained unlearning, aiming to remove both knowledge, memorization and privacy influence of news articles and copyrighted books. *Open-ended unlearning* tasks do not target the removal of specific training data; instead, they aim to erase broader concepts or behaviors without access to a defined forget corpus. WMDP involves a safety-alignment focus, targeting which targets undesired behaviors from hazardous knowledge related to curated datasets [Li et al., 2024b]. RWKU [Jin et al., 2024] and *Who's Harry Potter* (WHP) task [Eldan and Russinovich, 2023] require forgetting all knowledge related to famous entities. While benchmarks like TOFU, MUSE, PISTOL, LUME, and KnowUndo involve creating task models by injecting new knowledge via finetuning with the forget dataset; WMDP, RWKU and WHP [Eldan and Russinovich, 2023] operate directly on off-the-shelf LLMs to remove existing influence.

Unlearning evaluations: Each benchmark task involves multiple evaluations metrics that judge for unlearning success and for general utility preservation. These range from simple probability judgements in TOFU, to MIA-attack based metrics in MUSE, with dozens of metrics across benchmarks in the literature. Evaluating unlearning success is difficult, with several subsequent works questioning the reliability of benchmark metrics in various aspects [Kim et al., 2025, Lynch et al., 2024, Wang et al., 2024a, Doshi and Stickland, 2024, Zhang et al., 2025b].

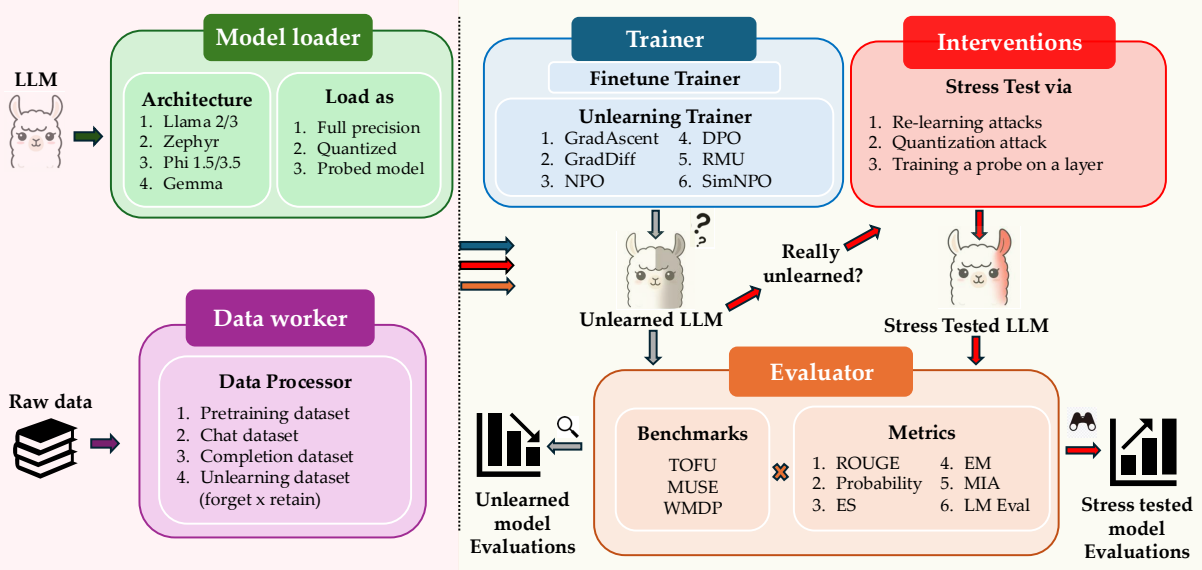


Figure 9.1: OU is an extensible library for benchmarking LLM unlearning methods and metrics. It provides a unified framework for implementing unlearning methods, unlearning metrics, and stress-testing tools to verify unlearning robustness. This figure illustrates the unlearning pipeline in terms of implementation-level components.

9.3 OpenUnlearning

The significant volume of research in LLM unlearning lacks unification both in technical implementations and in both unlearning method implementation and unlearning evaluation methodology. Existing benchmarks are implemented with a structure that makes it difficult to integrate with newer ones, hindering their adoption, and creating barriers to reproducibility that slow down progress. Moreover, unlearning methods and evaluation metrics aren't consistently extended across benchmarks, preventing standardization and comprehensive comparative analysis. We give a few examples of this fragmentation that cover key parts of the unlearning pipeline, from unlearning algorithms, to data processing, and evaluations,

1. **Fragmented evaluations of methods:** New methods are not implemented in all benchmarks. For example: UNDIAL [Dong et al., 2025] is not implemented on any of TOFU, MUSE and WMDP; NPO [Zhang et al., 2024d] is implemented with a different formulation for TOFU v/s MUSE; RMU was introduced only for WMDP etc. Similarly, evaluation metrics like MIA from MUSE [Shi et al., 2025] are not implemented in TOFU; and LM Eval Harness benchmarks used in WMDP can be extended to TOFU, MUSE.
2. **Disparate implementations of core components:** Several approaches involve customized loss functions [Zhang et al., 2024d, Maini et al., 2024a, Fan et al., 2024, Dong et al., 2025] and others make adjustments to optimization steps [Wang et al., 2025b, Jia et al., 2024, Fan

et al., 2025]. These techniques could be modularized and reused across tasks for deeper investigation and a fair comparison. Evaluation metrics use many common functionalities which can be shared across metric implementations (eg. probability, ROUGE-score and MIA statistics). Dataset pre-processing is separately implemented across datasets and benchmarks, while there are many common data types: like the pre-training corpora in WMDP and MUSE, and chat-style prompts in TOFU and RWKU. Some works have proposed stress tests for assessing the robustness of unlearning which could easily be a common feature across benchmarks.

To address this, we introduce OU: a unified, extensible pipeline that consolidates benchmarks, methods, evaluation metrics, datasets, and stress-tests under one roof (see Figure 9.1) to streamline unlearning implementations, benchmarking, and accelerate research.

9.3.1 Design of OpenUnlearning

Figure 9.1 gives an overview of OU's components. Our framework is designed with ease-of-use and easy extensibility in mind. All features are implemented in a structured, modular fashion, simplifying the process for researchers to integrate new datasets, evaluation metrics, unlearning methods, and entire benchmarks. Hydra [Yadan, 2019] is used for configuration management, with `YAML` files specifying each pipeline component and experiment parameters. This helps users effortlessly swap in modules and easily launch an experiment with a single command. A variety of modules, including model-loaders, trainers, dataset preprocessors, evaluation suites, evaluation metrics, experiment types and stress-test interventions are joined together in OU (listed in Table 9.1).

9.3.2 Design of modules

The procedure of extending OU with a new module variant generally involves two simple steps. (1) **Create and register a handler.** The `Python` class or function encapsulating the component's logic is implemented then registered to be accessed via a string key. (2) **Create the config.** The configuration `YAML` file names the handler key and specifies its parameters. Figure 9.2 provides an example illustrating this procedure for a new unlearning method.

Features: We currently support 9 unlearning algorithms, 6 model architectures, and 5 datasets ranging from chat to pretraining. Among existing benchmarks, we focus on the three most cited and used TOFU [Maini et al., 2024a], MUSE [Shi et al., 2025], WMDP [Li et al., 2024b] benchmarks. The framework includes a diverse set of metrics to assess model performance, including 16 unlearning metrics from existing benchmarks, as well

(a) Method implementation leveraging HuggingFace Trainer, followed by registration.

```
from transformers import Trainer
class Unlearner(Trainer):
  def compute_loss(self, ...):
    ...
  def get_optimizer_cls_and_kwargs(...)
    # custom optimizer
    ...
  def _inner_training_loop(self, ...):
    # modify training logic
    ...
_register_trainer(Unlearner)
```

(b) Configuration: create a YAML config specifying Training args and method parameters.

```
handler: Unlearner # map registered name

args: # HuggingFace Trainer args
num_epochs: 10
  learning_rate: 1e-5
  optim: shampoo

method_args:
  alpha: 1.0
  switch_every_n: 10
  retain_loss_type: NLL
```

Figure 9.2: Illustration of implementing a hypothetical unlearning method in OU

as additional evaluations by integrating LM Eval Harness [Gao et al., 2024c]. We also support three stress-testing approaches, which are essential for testing the robustness of unlearning, usually critical for model-owners in verifying compliance. All these features are summarized in Table 9.1 by component and variant. Our integration enriches each benchmark by enabling the use of metrics originally developed for others. For example, PrivLeak, initially introduced in MUSE, is now available in TOFU. More details on these technical benchmark improvements can be found in Appendix D. We also encourage community contributions by providing detailed guidelines for adding new benchmarks, unlearning methods, and evaluation metrics. This has already resulted in contributions from the community, with implementations for works like [Dong et al., 2025, Wang et al., 2025b, Yang, 2025].

OU is a living framework, and our design choices are built keeping easy integration of new components in mind. For instance, since the public release of our repository (with just TOFU and MUSE benchmarks) we introduced the WMDP benchmark, unlearning methods like RMU [Li et al., 2024b], UNDIAL [Dong et al., 2025], AltPO [Mekala et al., 2025]; evaluations like ES [Carlini et al., 2021c], EM [Tirumala et al., 2022], MIA [Duan et al., 2024a] and integrated evaluations like MUSE's PrivLeak (into TOFU) and LM Eval Harness [Gao et al., 2024c] (to enable WMDP evaluation) among many others. Additionally, we encourage community contributions by providing detailed guidelines for adding new benchmarks, unlearning methods, and evaluation metrics. This has already resulted in contributions from the community, with implementations for works like [Dong et al., 2025, Wang et al., 2025b, Yang, 2025]. Currently, each module supports several variants, with 3

popular LLM unlearning benchmarks, 5 task datasets, 9 unlearning methods, 16 evaluation metrics, 6 LLM architectures and 3 stress-tests.

9.4 Evaluating Unlearning Evaluations

Reliable evaluations for unlearning are essential for regulatory compliance and data privacy, yet remain challenging [Kim et al., 2025, Schwarzschild et al., 2024c, Łucki et al., 2025], especially for LLMs, due to ambiguity between memorization and generalization. We propose two minimal necessary desiderata—*Faithfulness* and *Robustness*—guided by our meta-evaluation framework, to promote trustworthy unlearning metrics (Figure 9.3).

Our meta-evaluation uses a test-bed of models with known ground truths to objectively assess metrics. We employ the TOFU benchmark [Maini et al., 2024a] with the improvements described from Appendix D with the `forget10` unlearning task (forgetting 10% of TOFU) comprising 400 examples. We use the LLaMA-3.2 1B model [Grattafiori et al., 2024], analyzing 12 unlearning metrics adjusted to $[0, 1]$ scale (see Appendix D).

9.4.1 Faithfulness

Faithfulness

Motivation. Unlearning evaluations may not faithfully reflect an LLM's knowledge. **Desideratum.** A faithful metric accurately reflects the presence of targeted knowledge by assigning consistently higher scores to models possessing it than to those lacking it.

LLMs often fail to regurgitate facts that remain encoded in their parameters when prompted, making it hard to tell whether a model truly forgot a target fact or simply refrained from exposing it [Doshi and Stickland, 2024, Lynch et al., 2024, Scholten et al., 2024, Wang et al., 2025a]. For example, work by Doshi and Stickland [2024] shows that simple paraphrasing of inputs can yield a tenfold increase in evaluation scores on 'unlearned' models, indicating that the apparent forgetting may only be superficial. "Deeper" evaluation metrics aim to quantify this knowledge more faithfully, like Truth Ratio [Maini et al., 2024a], GCG [Gandikota et al., 2024], or by using prompt engineering [Wang et al., 2025a, Shostack, 2024, Thaker et al., 2025].

On the other hand, evaluation metrics can register misleadingly high scores without the presence of the target knowledge [Maini et al., 2024a]. For example, in a question-answering evaluation using a simple ROUGE score, a model might achieve a high score by matching the parts of the target unrelated to the target fact. This calls for metrics that are

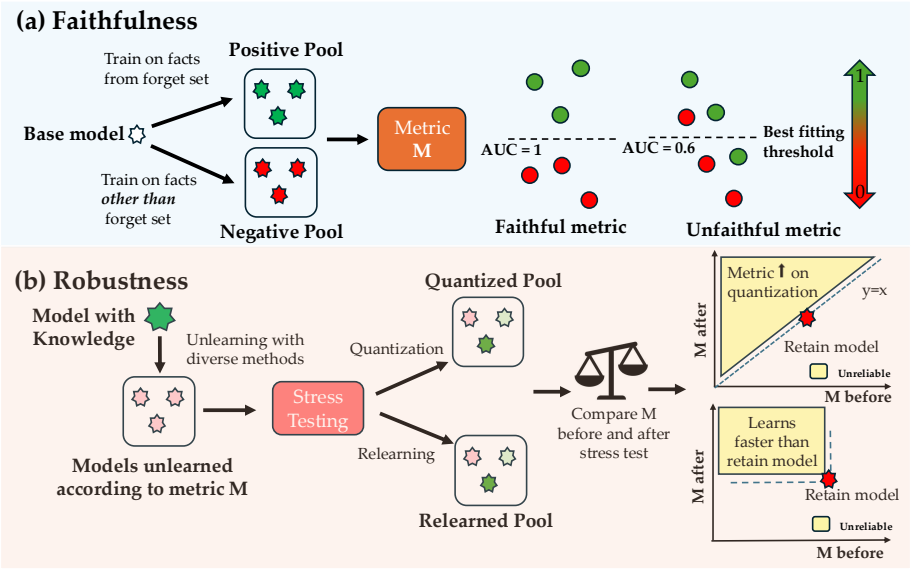


Figure 9.3: Meta-evaluation of unlearning metrics: (1) *Faithfulness*: the metric distinguishes models with and without target knowledge, reflected by high AUC; (2) *Robustness*: the metric value does not increase under benign changes (e.g., quantization) and does not improve faster than a retain model under non-benign changes (e.g., relearning).

faithful to the knowledge encoded in the model weights.

We measure *faithfulness* as the ability of metrics to distinguish between models trained with the forget dataset's knowledge (the *positive pool*, P) and those trained without it (the *negative pool*, N): (i) Each pool has 30 diverse models trained under varying conditions. (ii) These variants present the target `forget10` information for pool P models in diverse, challenging formats (e.g., biography vs. QA, paraphrases). Pool N models serve as negative controls, using similarly structured data lacking this target information using various perturbations and alternative datasets. (iii) Metric scores yield two distributions: $m(P)$, $m(N)$ (for P and N), and we compute AUC-ROC to quantify their separability. (iv) We select a classification threshold optimizing accuracy, which is subsequently used in robustness tests.

$$\text{Faithfulness} = \text{AUC-ROC}(m(P), m(N)) \tag{9.1}$$

9.4.2 Robustness

Robustness

Motivation. Unlearning evaluations can be vulnerable to stress-testing interventions. **Desideratum.** A robust metric's positive assessment of unlearning should (1) not flip upon benign model interventions; and (2) behave comparably to a model truly unfamiliar with the data under non-benign interventions.

Robustness of unlearning metrics is probed using various stress-test interventions. These include (1) relearning attempts, where the unlearned model is further trained to potentially recover the forgotten information [Lynch et al., 2024, Hu et al., 2025, Łucki et al., 2025, Wang et al., 2025a]; (2) information extraction via manipulating the model's internal representations [Belrose et al., 2023, Lynch et al., 2024, Seyitoğlu et al., 2024, Wang et al., 2025a, Arditi and Chughtai, 2024]; and (3) applying techniques like quantization [Zhang et al., 2025b]. Benign interventions, such as model quantization or relearning on non-forget data, do not reintroduce the forgotten knowledge. In contrast, non-benign interventions—like relearning directly on the forget set—explicitly re-expose the model to the targeted data. These stress tests have revealed that several unlearning evaluation metrics may be unreliable, often signaling successful unlearning even when the underlying knowledge remains recoverable.

For example, Zhang et al. [2025b] show that the PrivLeak metric [Shi et al., 2025] that previously reported a model as successfully unlearned can effectively 'flip' after a benign intervention, revealing that the targeted knowledge was perhaps never truly erased [Zhang et al., 2025b]. Such significant fluctuations under stress tests undermine the reliability of evaluation metrics. Furthermore, models unlearned with respect to a metric can exhibit high susceptibility on metric evaluation to non-benign interventions like relearning, where evaluation metrics show an unusually rapid return of the supposedly forgotten knowledge even with minimal retraining effort [Fan et al., 2025, 2024]. Robustness assesses stability under interventions such as relearning, probing and quantization. While probing was previously used by Wang et al. [2025a], Seyitoğlu et al. [2024], Lynch et al. [2024] to stress-test unlearning, in our setup, we found that probed models perform very poorly, with low scores across all metrics and show little discernible trends. Some probing results are shown in Appendix D.

Robustness to Relearning: We evaluate metric scores before (m^a) and after (m^b) relearning on forget-set data. Then, we compare relative metric score recovery rates between

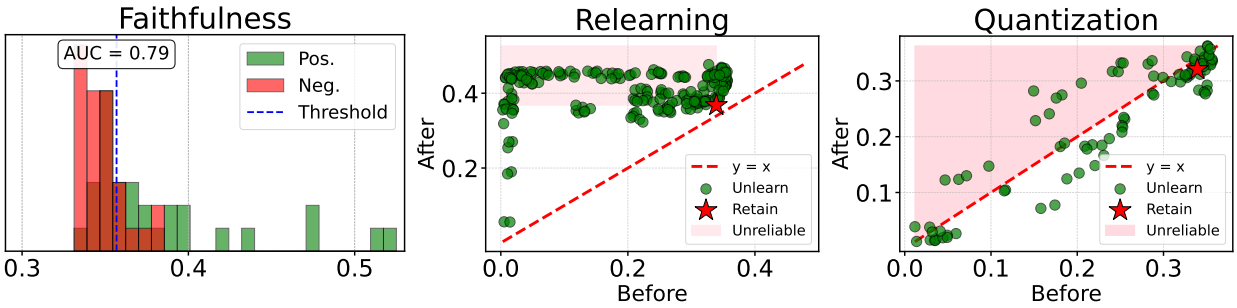


Figure 9.4: For the ROUGE metric we evaluate faithfulness (left) and robustness to quantization (middle), and relearning (right). Faithfulness achieves an AUC of 0.79, indicating substantial prediction overlap between models trained with and without the target knowledge. Relearning robustness is 0.48, showing many unlearned models re-acquire knowledge faster than the retain model upon re-exposure. Quantization robustness is 0.93, reflecting no distinctive trend of metric spikes post-quantization.

unlearned (m_{unl}) and retain (m_{ret}) models, where higher R implies greater robustness.

$$r = \frac{m^a_{\text{ret}} - m^b_{\text{ret}}}{m^a_{\text{unl}} - m^b_{\text{unl}}}, \quad R = \min(r, 1). \tag{9.2}$$

Robustness to Quantization: We quantize models to 4-bit precision and compute scores before and after quantization, where higher Q implies greater robustness.

$$q = \frac{m^b_{\text{unl}}}{m^a_{\text{unl}}}, \quad Q = \min(q, 1). \tag{9.3}$$

Realistic Model Filtering

We enforce practical constraints by filtering models with: (i) Utility drops exceeding 20%. (ii) Insufficient unlearning w.r.t. the considered metric (more than the threshold computed in Sec. 9.4.1's *faithfulness* analysis). Models which exhibit substantial model utility drops are unusable in practice and thus unlikely to inform robustness. Additionally, models that aren't unlearned w.r.t a metric are uninteresting for robustness analysis, since they do not reflect realistic scenarios where some kind of unlearning is observed before models are stress tested. The case of interest is when an ostensibly performant LLM exhibits low scores according to a chosen metric, indicating unlearning, and practitioners require confidence in the metric's judgement.

We analyze roughly 400 diverse models from various unlearning methods to reflect realistic use cases. We ensure diversity by using models unlearned using the GradDiff,

IdkDPO, IdkNLL [Maini et al., 2024a], NPO [Zhang et al., 2024d], SimNPO [Fan et al., 2024], AltPO [Mekala et al., 2025], UNDIAL [Dong et al., 2025] and RMU [Li et al., 2024b] unlearning methods (methods described in Appendix Sec. D and hyperparameters in Sec. D). This aligns the distributions between the unlearned model pools used in our analysis and unlearned models selected by practitioners.

9.4.3 Aggregation of Metrics

We consolidate evaluations through harmonic mean, ensuring balanced performance across criteria:

$$\text{Robustness} = \text{HM}(R, Q), \quad \text{Overall} = \text{HM}(\text{Faithfulness}, \text{Robustness}) \tag{9.4}$$

An effective unlearning metric must be both faithful in representing unlearning and robust in its measurements; a trivial constant-value metric, for instance, would be robust but entirely unfaithful. To holistically assess a metric, we aggregate these distinct qualities using the Harmonic Mean (HM), as this ensures that a high final score demands strong performance in all constituent parts. Figure 9.4 illustrates these distributions and scores for the ROUGE metric as an example. Further methodological considerations, including comparisons to prior work, are detailed in Appendix D.

9.4.4 Results and Discussion

Table 9.2 highlights key insights: (i) **Extraction Strength (ES)** [Carlini et al., 2021c] emerges as most reliable overall, aligning with Wang et al. [2025a]. (ii) **Truth Ratio** has superior faithfulness but lower robustness, ranking third overall. (iii) Metrics based on raw probabilities or ROUGE scores have moderate faithfulness and robustness, limiting their reliability. (iv) Membership inference (MIA)-based metrics demonstrate high faithfulness but lack robustness, cautioning against relying solely on MIA metrics for assessing unlearning. This sensitivity raises concerns about the reliability of the MIA-based privacy assessments in unlearning contexts as introduced by Shi et al. [2025], as even benign interventions can reverse unlearning effects, as observed in Zhang et al. [2025b].

Our extensive model testbed supports ongoing development of improved, practical unlearning metrics. Our testbed comprising 450+ models — including those from pools P, N, and various unlearned model checkpoints — offers a valuable platform for the creation and rigorous assessment of improved unlearning evaluation metrics. Metrics validated on this testbed can then be applied with greater confidence to real-world unlearning scenarios. Our overarching goal is to stimulate the development of more faithful and

trustworthy metrics, leveraging the insights from our meta-evaluation framework. This meta-evaluation setup can be expanded by incorporating more diverse unlearning setups, model architectures and newer methods. Newer adversarial model setups will be needed to challenge metrics as they improve on existing testbeds. Such a dynamic approach ensures that unlearning methods and their meta-evaluations can mutually inform each other, driving progress as unlearning research advances.

9.5 Benchmarking Unlearning Methods

Unlike prior works with limited baselines and metrics, OU provides a standardized and scalable framework to conduct a large-scale comparison of various unlearning methods. We demonstrate this by evaluating 8 unlearning methods using 10 evaluation metrics on TOFU.

Unlearning methods: OU enables evaluation across a broader range of methods, including SimNPO [Fan et al., 2024], RMU [Li et al., 2024b], AltPO [Mekala et al., 2025], NPO [Zhang et al., 2024d], UNDIAL [Dong et al., 2025], as well as baselines like IdkPO, IdkNLL, and GradDiff [Maini et al., 2024a]. See Appendix D for each method's definition.

Evaluation metrics: We evaluate unlearning methods using memorization metrics validated in our meta-analysis, alongside privacy and utility metrics. Using the TOFU benchmark, and following the SemEval 2025 LLM Unlearning Challenge's ranking procedure [Ramakrishna et al., 2025b], we compute a composite score by aggregating metrics from the three categories: memorization (using the 4 top-performing knowledge metrics from Sec. 9.4's metric meta-evaluation: ES, EM, Truth Ratio, Paraphrased Probability), privacy (4 MIA metrics), and utility (2 metrics, including TOFU's Model Utility and forget-set fluency). Exact details of our metric aggregation are in Appendix D. Note that the memorization score (reported in Table 9.3) corresponds to forgetting: higher Mem. indicates less knowledge.

Tuning strategy: To ensure fairness, 27 hyperparameter tuning trials are allocated per method, as tuning can significantly improve performance of even simple baselines [Wang et al., 2025a]. Due to the impracticality of tuning on privacy metrics, that require the presence of i.i.d. holdout datasets and oracle retain models (i.e., models trained solely on the retain set, with no exposure to the forget set), we validate models only on accessible metrics that capture memorization and utility. Additionally, model selection during tuning can significantly affect rankings (Appendix D).

Results and discussion: While memorization, privacy, and utility each capture a distinct aspect of unlearning quality, aggregating them using a harmonic mean (Table 9.3), results in SimNPO [Fan et al., 2024] ranking first. Although its memorization score trails that of others, it remains close to the retain model's level, avoiding over-unlearning. SimNPO fully preserves utility and achieves competitive privacy results, striking a balance across all three criteria. The next best performer is RMU, which demonstrates strong memorization and privacy but suffers a significant drop in utility.

Methods that under-unlearn (e.g. IdkNLL, which yields a low memorization score i.e. less forgetting) score lower on privacy. On the other hand, methods like GradDiff over-unlearn, forgetting too aggressively, yielding a high memorization score. This leads to poor privacy performance, as the model's behavior deviates significantly from that of the retain model. This suggests that detecting and halting unlearning once the model's behavior has reverted to its "default" state is crucial to ensure privacy.

Because different ranking schemes can produce very different rankings (Table 9.3 v/s Appendix Table D.3), it is critical to choose an appropriate method ranking procedure and aggregate metrics. Additionally, there is a lack of standardization on which metrics are suitable for model selection versus final evaluation (elaborated upon in Appendix D). While identifying the ideal ranking method and model selection approach is beyond our scope, we release all unlearned model checkpoints from our study to support future research on fair evaluation.

9.6 Conclusion

The field of LLM unlearning has faced challenges due to fragmented methodologies and inconsistent evaluations. To address this, we introduced OU, a standardized and extensible framework that unifies research efforts by integrating 9 unlearning algorithms, 16 evaluation metrics, and 3 major benchmarks. This comprehensive platform enabled us to conduct a novel meta-evaluation of unlearning metrics, assessing their faithfulness and robustness, and to perform large-scale benchmarking of unlearning methods. Our meta-evaluation identified Extraction Strength (ES) and Exact Memorization (EM) as particularly reliable metrics, with Truth Ratio also showing high faithfulness. Benchmarking revealed SimNPO and RMU as strong performers, though we also observed significant sensitivities in ranking. At the same time OU, by providing a common ground and releasing numerous model checkpoints, establishes a clear pathway for the community towards more rigorous, reproducible, and accelerated development of robust unlearning techniques and evaluation protocols, ultimately fostering safer AI deployments.

Component		Variants
Models		LLaMA-2, 3.1, 3.2 [Touvron et al., 2023, Grattafiori et al., 2024] Zephyr-7B [Tunstall et al., 2024] Phi-1.5, 3.5 [Li et al., 2023f, Abdin et al., 2024a]
Unlearning algorithms		GradAscent, GradDiff, IdkDPO, IdkNLL [Maini et al., 2024a] NPO [Zhang et al., 2024d] SimNPO [Fan et al., 2024] RMU [Li et al., 2024b] UNDIAL [Dong et al., 2025] AltPO [Mekala et al., 2025]
Datasets		TOFU: bios [Maini et al., 2024a] WMDP: cyber, bio [Li et al., 2024b] MUSE: news, books [Shi et al., 2025]
Evaluation suites		TOFU [Maini et al., 2024a] MUSE [Shi et al., 2025] WMDP [Li et al., 2024b] LM Eval [Gao et al., 2024c]
Metrics	**Mem.**	Verbatim Prob. / ROUGE [Maini et al., 2024a, Shi et al., 2025] Knowledge QA- ROUGE [Maini et al., 2024a, Shi et al., 2025] Extraction Strength [Carlini et al., 2021c] Exact Memorization [Tirumala et al., 2022]
	Privacy	Forget Quality [Maini et al., 2024a] LOSS [Yeom et al., 2018b] ZLib [Carlini et al., 2021c] GradNorm [Wang et al., 2024a] MinK [Shi et al., 2023b] MinK++ [Zhang et al., 2025a] Privacy Leakage [Shi et al., 2025]
	Utility	Truth Ratio, Model Utility [Maini et al., 2024a] LM-Eval [Gao et al., 2024c] (WMDP, MMLU, etc.) Fluency [Mekala et al., 2025]
Stress tests		Relearning [Hu et al., 2025, Lynch et al., 2024, Łucki et al., 2025, Wang et al., 2025a] Quantization [Zhang et al., 2025b] Probing [Lynch et al., 2024, Seyitoğlu et al., 2024

Table 9.1: Overview of existing OU components and their available feature variants. The design is easily extensible, allowing users to seamlessly contribute new features.

Metrics	Agg. ↑	Faithful. ↑	Robustness ↑		
			Agg. ↑	Quant. ↑	Relearn ↑
Extraction Strength	**0.85**	0.92	**0.79**	**0.95**	0.68
Exact Mem.	<u>0.80</u>	0.90	0.72	0.92	0.59
Truth Ratio	0.73	**0.95**	0.59	0.92	0.43
Para. Prob.	0.73	0.71	<u>0.75</u>	0.60	**0.98**
Para. ROUGE	0.72	0.89	0.61	<u>0.93</u>	0.45
Probability	0.72	0.82	0.65	0.60	<u>0.70</u>
ROUGE	0.70	0.79	0.64	<u>0.93</u>	0.48
Jailbreak ROUGE	0.69	0.83	0.59	0.85	0.45
MIA - ZLib	0.71	0.92	0.57	0.56	0.59
MIA - MinK	0.67	<u>0.93</u>	0.52	0.48	0.57
MIA - LOSS	0.66	<u>0.93</u>	0.52	0.48	0.57
MIA - MinK++	0.61	0.81	0.48	0.61	0.40

Table 9.2: Meta-evaluation of 12 unlearning metrics for Faithfulness and Robustness. Robustness is assessed using two stress-testing methods: quantization and relearning, with their harmonic mean reported as Agg. An overall aggregation across both Faithfulness and Robustness is reported in the first Agg. column. Higher scores indicate better performance (↑) in all dimensions. The best values are shown in bold, and the second-best values are underlined.

Method	Agg. ↑	Mem. ↑	Priv. ↑	Utility ↑
Init. finetuned	0.00	0.00	0.10	1.00
Retain	0.58	0.31	1.00	0.99
SimNPO [Fan et al., 2024]	**0.53**	0.32	**0.63**	**1.00**
RMU [Li et al., 2024b]	<u>0.52</u>	0.47	<u>0.50</u>	0.61
UNDIAL [Dong et al., 2025]	0.42	0.27	0.48	0.78
AltPO [Mekala et al., 2025]	0.15	<u>0.63</u>	0.06	0.95
IdkNLL [Maini et al., 2024a]	0.15	0.08	0.17	0.93
NPO [Zhang et al., 2024d]	0.15	0.52	0.06	<u>0.99</u>
IdkDPO [Maini et al., 2024a]	0.14	0.56	0.06	0.95
GradDiff [Maini et al., 2024a]	9e-3	**0.97**	3e-3	0.79

Table 9.3: Comparison of unlearning methods on the TOFU task, showing overall aggregate (Agg.), memorization (Mem.), privacy (Priv.), and utility (Utility) scores. Higher scores indicate better performance (↑). Initial finetuned is the target model before unlearning and Retain model is the gold standard target model. The best values are shown in bold, and the second-best values are underlined.

Chapter 10

SSFT: Second-Split Forgetting Time

Based on Pratyush Maini, Saurabh Garg, Zachary C. Lipton, J. Zico Kolter. NeurIPS 2022

Abstract

We study split-specific dynamics of learning and forgetting, introducing SSFT as an indicator of how models retain or forget content across training phases.

10.1 Introduction

A growing literature has investigated metrics for characterizing the difficulty of training examples, driven by such diverse motivations as (i) deriving insights for how to reconcile the ability of deep neural networks to generalize [Krizhevsky et al., 2012] with their ability to memorize noise [Zhang et al., 2021a, Feldman, 2020a]; (ii) identifying potentially mislabeled examples; and (iii) identifying notably challenging or rare sub-populations of examples. Some of these efforts have turned towards learning dynamics, with researchers noting that neural networks tend to learn cleanly labeled examples before mislabeled examples [Liu et al., 2020, Frankle et al., 2020b, Garg et al., 2021], and more generally tend to learn *simpler* patterns sooner—for several intuitive notions of simplicity [Shah et al., 2020, Mangalam and Prabhu, 2019, Hacohen et al., 2020]. Broadly, works in this area tend to characterize examples as belonging either to *prototypical groups* or *memorized exceptions* [Feldman and Zhang, 2020, Jiang et al., 2020, Carlini et al., 2019a]. Adapting these intuitions to real datasets, Feldman [2020a] propose rating the degree to which an example is memorized based on whether its predicted class flips when it is excluded from the training set. These, and other works [Chatterjee, 2020, Mangalam and Prabhu, 2019, Shah et al., 2020, Hooker et al., 2019, Toneva et al., 2018] have proposed many metrics for characterizing example difficulty with Carlini et al. [2019a] comparing five such metrics. However, while many of these works distinguish some notion of *easy* versus *hard* samples, they seldom (i) offer tools for distinguishing among different types of hard examples; (ii) explain theoretically why these metrics might be useful for distinguishing easy versus hard samples. Moreover,

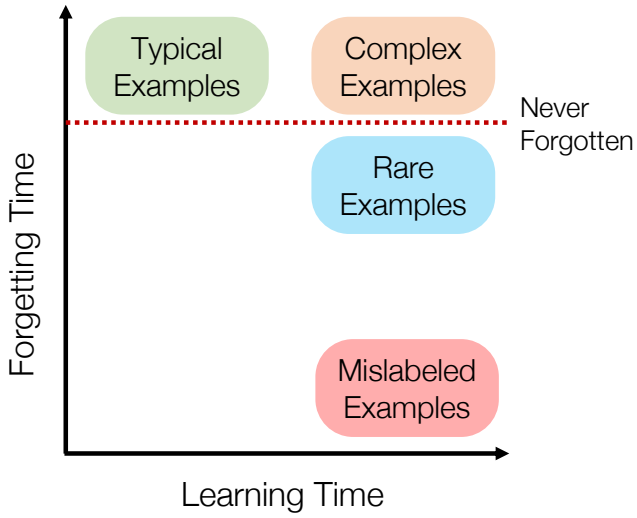


Figure 10.1: Overview of example separation offered by the unified view of learning and forgetting time.

existing metrics tend to give similar scores to examples that are difficult for distinct reasons, e.g, membership in rare, complex, or mislabeled sub-populations.

In this paper, we propose to additionally consider a new metric, Second-Split Forgetting Time (SSFT), calculated based on the forgetting dynamics that arise as training examples are forgotten when a neural network continues to train on a second, randomly held out data partition. SSFT is defined as the fine-tuning epoch after which a first-split training example is no longer classified correctly. We find that SSFT identifies mislabeled examples remarkably well but does little to separate out under- versus over-represented subpopulations. Conversely, metrics based on the (first-split) training dynamics are more discriminative for separating these populations but less useful for detecting mislabeled examples. We leverage the complementarity of first- and second-split metrics, showing that by jointly visualizing the two, we can produce a richer characterization of the training examples.

In our experiments, we operationalize several notions of hard examples, namely: (i) **mislabeled** examples, for which the original label has been flipped to a randomly chosen incorrect label; (ii) **rare** examples, which belong to underrepresented subpopulations; and (iii) **complex** examples, which belong to subpopulations for which the classification task is more difficult (details in Section 10.3.2). We perform specific ablation studies with datasets complicated by just one type of hard example (Section 10.4.3), and show how SSFT can help to distinguish among these categories of examples. We observe that during second-split training, neural networks (i) first forget mislabeled examples from the first split; (ii)

only slowly begin to forget *rare* examples (e.g., from underrepresented sub-populations) unique to the first training set; and (iii) do not forget complex examples.

This separation of hard example types has multiple practical applications. **First**, we can use the method to identify noisy labels: On CIFAR-10 with 10% added class noise, SSFT achieves 0.94 AUC for identifying mislabeled samples, while the first-split metrics range in AUC between 0.58 to 0.90. **Second**, the method can also help improve generalization in noisy data settings: while the removal of hard examples according to first-split learning time degrades the performance of the classifier, the removal of hard examples according to SSFT can actually *improve* generalization. This is especially beneficial when e.g., training on synthetic data (produced by a generative model) or mislabeled data. **Third**, we show how SSFT can identify failure modes of machine learning models. For example, in a simplified task classifying between horses and airplanes in the CIFAR-10 dataset, we find that training examples containing horses with sky backgrounds and airplanes with green backgrounds are among the earliest forgotten—indicating that the model relies on the background as a spurious feature. **Last**, we also add that our metric is robust across multiple seeds, and the earliest forgotten examples are robust across architectures. Across multiple optimizers, SSFT distinguishes mislabeled samples, whereas first-split metrics appear more sensitive to the choice of optimizer.

Finally, we investigate second-split dynamics theoretically, analyzing overparametrized linear models [Soudry et al., 2018]. We introduce notions of mislabeled, rare, and complex examples appropriate to this toy model. Our analysis shows that mislabeled examples from the first split are forgotten quickly during second-split training whereas rare examples are not. However, as we train for a long time, rare examples from the first split are eventually forgotten as the model converges to the minimum norm solution on the second split while predictions on complex examples remain accurate with high probability.

10.2 Theoretical Results

Through our theoretical analysis, we will characterize the forgetting dynamics of mislabeled, rare and complex examples in a simplified version of the framework used for our synthetic experiments in Figure 10.2a. Recall, our setup contains two dataset splits $\mathcal{D}_A, \mathcal{D}_B$, where we train on the first split until achieving perfect accuracy on all training points, and then with these weights train on $\mathcal{D}_B$ for infinite time. In particular, we will prove that both mislabeled and rare examples are forgotten upon training for infinite time, with mislabeled examples being forgotten much faster. Further, we will show that complex examples from the first split do not get forgotten if not continually trained on. We assume in our analysis

that $\mathcal{D}_B$ has no mislabeled or rare examples, and $\mathcal{D}_A$ contains one example of each type.

We consider a dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i\}^n$ such that $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{X} \times \mathcal{Y}$, and $\mathbf{x}_i = \boldsymbol{\mu}_g + \mathbf{z}_i$ where $\mathbf{z}_i \sim \mathcal{N}\left(0, \sigma^2 \mathbf{I}_d\right)$, and $\left\|\boldsymbol{\mu}_g\right\|_2^2 = k\mu^2$ (as in Section 10.4.2). Let $\mathbf{w} \in \mathbb{R}^d$ represent the weight vector of an overparametrized linear model. We analyze the learning and forgetting dynamics by minimizing the empirical risk: $\mathcal{L}(\mathcal{D}; \mathbf{w}) = \sum_i \ell(\mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i)$, where ℓ is the exponential loss. Following Chatterji and Long [2021], we make the following assumptions about the problem setup:

(A.1) The failure probability satisfies $0 \leq \delta \leq 1/C$,

(A.2) The number of samples satisfies $n \geq C \log(1/\delta)$,

(A.3) The input dimension $d \geq C \max\{n^2 \log(n/\delta), n(k \cdot \mu^2/\sigma^2)\}$, and $k \cdot \mu^2/\sigma^2 \geq C \log(n/\delta)$,

where $k \cdot \mu^2/\sigma^2$ represents the signal to noise ratio and C is a large constant. Now we formalize the notions of rare, mislabeled and complex examples for our theoretical analysis.

Definition 1 (Rare Examples, $\mathcal{A}$ [Feldman, 2020a])**.** *Consider a dataset $\mathcal{D}$ sampled from a mixture of distributions $\{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ with frequency $\{\pi_1, \ldots, \pi_N\}$ respectively. Let $\mathcal{A} \subseteq \mathcal{D}$ be the set of rare examples. Then, for all $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{A}$, if $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_j$, then there are $O(1)$ samples from $\mathcal{D}_j$ in $\mathcal{D}$.*

Definition 2 (Mislabeled Examples, $\mathcal{M}$)**.** *Consider a k class classification problem with $\mathcal{D}_Y = \{1, 2, \ldots, k\}$. Let $\mathcal{M} \subset \mathcal{D}$ be the set of mislabeled examples. Then for any $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}$, a corresponding mislabeled example is given by $(\mathbf{x}, \tilde{\mathbf{y}}) \in \mathcal{M}$ such that $\tilde{\mathbf{y}} \in \mathcal{D}_Y \setminus \{\mathbf{y}\}$.*[1]

Definition 3 (Complex Examples, C)**.** *Let $C \subset \mathcal{D}$ be the set of examples sampled from complex distributions. Let $(\mathbf{x}_i, \mathbf{y}_i) \in C$ such that $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_g$ (complex group), then $\mu_g = \frac{\mu_t}{\lambda}$, $\lambda > 1$ where μ_t is the coordinate-wise mean for samples drawn from any simple distribution $\mathcal{D}_t$ (Section 10.4.2).*

Optimization We perform gradient descent with fixed learning rate η,

$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla \mathcal{L}(\mathbf{w}(t)) = \mathbf{w}(t) - \eta \sum_i \ell'(\mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i) \cdot \mathbf{y}_i \mathbf{x}_i. \tag{10.1}$$

Solution dynamics For sufficiently small learning rate η, and (bounded) starting point $\mathbf{w}(0)$, Soudry et al. [2018] showed that:

$$\mathbf{w}(t) = \hat{\mathbf{w}} \log t + \rho(t), \tag{10.2}$$

[1] For binary classification, $\mathcal{D}_Y = \{-1, +1\}$. The labels are reversed for mislabeled examples.

where $\rho(t)$ is a bounded residual term, and $\hat{\mathbf{w}}$ is the solution to the hard margin SVM:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{w}\|_2^2 \quad s.t.\ \mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i \geq 1, \tag{10.3}$$

10.2.1 First-split Learning

For stage 1, we consider that we train the model for a maximum of T epochs (until we achieve 100% accuracy on the first training dataset $\mathcal{D}_A$). This means that the learned weight vectors are close to, but have not converged to the max margin solution. The solution at the end of t epochs is given by $\mathbf{w}_A(t)$. At sufficiently large T, we have:

$$\begin{aligned} \mathbf{w}_A(T) &= \hat{\mathbf{w}}_A \log T + \rho_A(T) \\ \mathbf{w}_A(T)^\top \mathbf{y}_i \mathbf{x}_i &\geq 1 \quad \forall (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_A \end{aligned} \tag{10.4}$$

10.2.2 Second-split Forgetting

We initialize the weights for second stage of training with $\mathbf{w}_A(T)$ from first training stage, and then train on $\mathcal{D}_B$. We provide the formal theorem statement and complete proofs in Appendix E, but provide informal theorem statements and an intuitive proof sketch below:

Theorem 3 (Asymptotic Forgetting (informal))**.** *For sufficiently small learning rate, given datasets $\mathcal{D}_A, \mathcal{D}_B \sim \mathcal{D}^n$. After training for $T' \to \infty$ epochs, the following hold with high probability:*

1. *Mislabeled and Rare examples from $\mathcal{D}_A$ are forgotten.*
2. *Complex examples from $\mathcal{D}_A$ are not forgotten.*

Proof Sketch. We use the result from Soudry et al. [2018] that for any bounded initialization, when trained on a separable data, the model converges to the same min-norm solution. As a result, we can ignore the impact of $\mathcal{D}_A$ at infinite time training. Then, we use generalization bounds from Chatterji and Long [2021] to argue about the accuracy on mislabeled and complex examples. For the case of rare examples, we show that the probability of correct model prediction can be approximated by a Gaussian CDF with mean 0 and $O(1/\sqrt{n})$ variance.

Theorem 4 (Intermediate-Time Forgetting (informal))**.** *For sufficiently small learning rate, given two datasets $\mathcal{D}_A, \mathcal{D}_B \sim \mathcal{D}^n$. For a model initialized with weights, $\mathbf{w}_B(0) = \mathbf{w}_A(T)$ and trained for $T' = f(T)$ epochs, the following hold with high probability:*

1. *Mislabeled examples from $\mathcal{D}_A$ are no longer incorrectly predicted.*

2. *Rare examples from $\mathcal{D}_A$ are not forgotten.*

Proof Sketch. $\mathcal{D}_B$ contains examples from the same majority distributions as $\mathcal{D}_A$. The mislabeled example also belongs to one of these distributions, but has the opposite label. However, $\mathcal{D}_B$ does not have samples from rare groups found in $\mathcal{D}_A$. Using representer theorem, we decompose the model updates into a weighted sum of each training data point in $\mathcal{D}_B$. Then, we analyze the change in prediction on rare and mislabeled examples, which is a dot product of the weight update with $\mathbf{x}_m$ or $\mathbf{x}_r$. Per our assumptions, the the mean of each group $\boldsymbol{\mu}_g$ is orthogonal to the other. As a result, the rare example finds negligible coupling with any example in $\mathcal{D}_B$, and the variance of its prediction keeps increasing due to the noise term contributed in the model weights from each example in $\mathcal{D}_B$. On the contrary, the mislabeled examples have a strong coupling with all the examples in its group. Due to its incorrect label, the mean of its predictions moves towards the correct label, with variance increasing at a similar rate. The final step is to jointly analyze the rate of change of prediction of both the examples, and find an optimal time T' when the prediction on the mislabeled example is flipped and the rare example still retains its prediction with high probability.

10.3 Method

The primary goal of our work is to *characterize* the hardness of different datapoints in a given dataset. Suppose we have a dataset $\mathcal{D}_A = \{\mathbf{x}_i, \mathbf{y}_i\}^n$ such that $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}$. For the purpose of characterization, we augment each datapoint $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_A$ with parameters (t_i, τ_i) where t_i quantifies the first-split learning time (FSLT), and τ_i quantifies the second-split forgetting time (SSFT) of the sample. To obtain these parameters, we next describe our proposed procedure.

Procedure We train a model f on $\mathcal{D}$ to minimize the empirical risk: $\mathcal{L}(\mathcal{D}; f) = \sum_i \ell(f(\mathbf{x}_i), \mathbf{y}_i)$. We use f_A to denote a model f (initialized with random weights) trained on $\mathcal{D}_A$ until convergence (100% accuracy on $\mathcal{D}_A$). We then train a model initialized with f_A on a held-out split $\mathcal{D}_B \sim \mathcal{D}^n$ until convergence. We denote this model with $f_{A \to B}$. To obtain parameters (t_i, τ_i), we track per-example predictions $(\hat{\mathbf{y}}_i^t)$ at the end of every epoch (t^{th}) of training. Unless specified otherwise, we train the model with cross-entropy loss using Stochastic Gradient Descent (SGD).

Definition 8 (First-Split Learning Time)**.** *For $\{\mathbf{x}_i, \mathbf{y}_i\} \in \mathcal{D}_A$, learning time is defined as the earliest epoch during the training of a classifier f on $\mathcal{D}_A$ after which it is always classified correctly, i.e.,*

$$t_i = \arg\min_{t^*} (\hat{\mathbf{y}}_{i,(A)}^t = \mathbf{y}_i \;\; \forall t \geq t^*) \quad \forall \{\mathbf{x}_i, \mathbf{y}_i\} \in \mathcal{D}_A. \tag{10.5}$$

Definition 9 (Second-Split Forgetting Time). *Let $\hat{\mathbf{y}}^t_{i,(A\to B)}$ to denote the prediction of sample $\{\mathbf{x}_i, \mathbf{y}_i\} \in \mathcal{D}_A$ after training $f_{(A\to B)}$ for t epochs on $\mathcal{D}_B$. Then, for $\{\mathbf{x}_i, \mathbf{y}_i\} \in \mathcal{D}_A$ forgetting time is defined as the earliest epoch after which it is never classified correctly, i.e.,*

$$\tau_i = \arg\min_{t^*} (\hat{\mathbf{y}}^t_{i,(A\to B)} \neq \mathbf{y}_i \quad \forall t \geq t^*) \quad \forall \{\mathbf{x}_i, \mathbf{y}_i\} \in \mathcal{D}_A. \tag{10.6}$$

10.3.1 Baseline Methods

We provide a brief description of metrics for example hardness considered in recent comparisons [Jiang et al., 2020].

Number of Forgetting Events: (n_f). An example $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}$ undergoes a forgetting event when the accuracy on the example decreases between two consecutive updates. Toneva et al. [2018] analyzed the total number of such events n_f during the training of a neural network to identify hard examples.

Cumulative Learning Accuracy: (acc_l). Jiang et al. [2020] suggest that rather than using the learning time (Definition 8), using the number of epochs during training when a machine learning model correctly classifies a given sample is a more stable metric for predicting example hardness.

Cumulative Learning Confidence: (conf_l). Similar to acc_l, conf_l measures the cumulative softmax confidence of the model towards the correct class over the course of training.

10.3.2 Example Characterization

We characterize example hardness via three sources of learning difficulty: **(i) Mislabeled Examples:** We refer to mislabeled examples as those datapoints whose label has been flipped to an incorrect label uniformly at random. **(ii) Rare Examples:** We assume that rare examples belong to sub-populations of the original distribution that have a low probability of occurrence. In particular, there exist $O(1)$ examples from such sub-populations in a given dataset. In Section 10.4.3 we describe how we operationalize this notion in the case of the CIFAR-100 dataset. **(iii) Complex Examples:** These constitute samples that are drawn from sub-groups in the dataset that require either (1) a hypothesis class of high complexity; or (2) higher sample complexity to be learnt relative to examples from rest of the dataset. We leave the definition of complex samples mathematically imprecise, but with the same intuitive sense as in prior work [Shah et al., 2020, Arpit et al., 2017]. For instance, in a dataset composed of the union of MNIST and CIFAR-10 images, we would consider the subpopulation of CIFAR-10 images to be more *complex*.

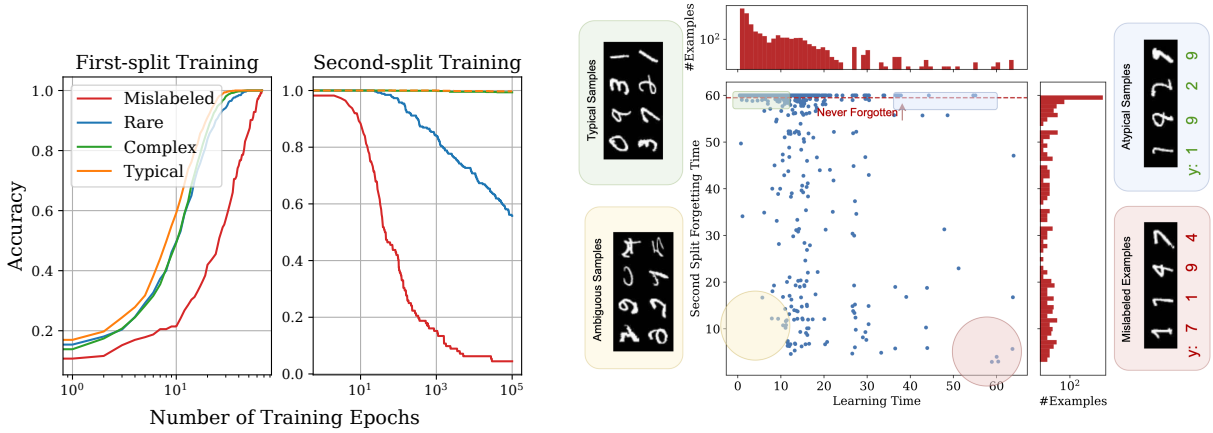


Figure 10.2: Rate of Learning and Forgetting of examples for different groups in the synthetic dataset. While first-split training is not able to distinguish between rare and complex examples, second-split training succeeds in distinguishing them. Additionally, second-split training separates mislabeled examples from the rest relatively better than first-split training. (b) Visualization of first-split learning and second-split forgetting times when training LeNet model on the MNIST dataset.

10.4 Empirical Investigation of First- and Second-Split Training Dynamics

10.4.1 Experimental Setup

Datasets We show results on a variety of image classification datasets—MNIST [Deng, 2012], CIFAR-10 [Krizhevsky et al., 2009], and Imagenette [Howard]. For experiments in the language domain, we use the SST-2 dataset [Socher et al., 2013]. For each of the datasets, we split the training set into two equal partitions ($\mathcal{D}_A, \mathcal{D}_B$). For experiments with mislabeled examples, we simulate mislabeled examples by randomly selecting a subset of 10% examples from both the partitions and changing their label to an incorrect class.

Training Details Unless otherwise specified, we train a ResNet-9 model [Baek, 2019] using SGD optimizer with weight decay 5e-4 and momentum 0.9. We use the cyclic learning rate schedule [Smith, 2018] with a peak learning rate of 0.1 at the 10th epoch. We train for a maximum of 100 epochs or until we have 5 epochs of 100% training accuracy. We first train on $\mathcal{D}_A$, and then using the pre-initialized weights from stage 1, train on $\mathcal{D}_B$ with the same learning parameters. All experiments can be performed on a single RTX2080 Ti. Complete hyperparameter details are available in Appendix E.

10.4.2 Learning-Forgetting Spectrum for various datasets

Synthetic Dataset We consider data $(\mathbf{x}, \mathbf{y})$ sampled from a mixture of multiple distributions $\mathcal{D}_g$, s.t. $\mathbf{x} \in \mathbb{R}^d$. $\mathcal{D}_g$ denotes the g^{th} group and has a sampling frequency of π_g. Each group $\mathcal{D}_g \equiv (\mathcal{D}_{X,g}, \{\mathbf{y}_g\})$, i.e., the true label for all the samples drawn from a given group is the same, and the examples in each group are non-overlapping. Each group is parametrized by a set of $kL^2(\Omega)d$ unique indices $\mathcal{I}_g \subset [d]$ such that $\mathcal{I}_i \cap \mathcal{I}_j = \phi$ for $i \neq j$. The discriminative characteristic of each group is the vector $\boldsymbol{u}_g$, such that, $[\boldsymbol{u}_g]_i = 1$ if $i \in \mathcal{I}_g$ else 0 $\forall i \in [d]$. Then for any sample $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}$:

$$P(\mathbf{x} \in \mathcal{D}_{X,g}) = \pi_g; \quad \mathbf{x}|\mathcal{D}_{X,g} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d) + \boldsymbol{\mu}_g.$$

For our simulation, we consider a 10 class-classification problem, with $\mu_g = 5$ for typical groups, and $\mu_g = 4$ for complex groups (higher signal to noise ratio). For any sample drawn from a rare group, we have $O(1)$ samples from that group in the entire dataset $(\mathcal{D}_A \cup \mathcal{D}_B)$. Mislabeled samples are only generated from the majority typical groups. In Figure 10.2a, we show the rate of learning and forgetting of examples from each of these categories. We note that in the second-split training, the mislabeled examples are quickly forgotten, and the complex examples are never forgotten. The rare examples are forgotten slowly. In Section 10.2 we will theoretically justify the observations in the synthetic dataset and show that the rare examples are expected to be forgotten as we train for an infinite time.

Image Domain In Figure 10.2b, we show representative examples in the four quadrants of the learning-forgetting spectrum. More specifically, we find that the examples forgotten fastest and learned last are mislabeled. And the ones learned early and never forgotten once learned are characteristic simple examples of the MNIST dataset. Examples in the first and third quadrant are seemingly atypical and ambiguous respectively. Similar visualizations for other image datasets can be found in Appendix E.

Other Modalities The forgetting and learning dynamics occur broadly across modalities apart from images. We repeat the same problem setup on the SST-2 [Socher et al., 2013] dataset for sentiment classification. We fine-tune a pre-trained BERT-base model [Devlin et al., 2018] successively on two disjoint splits of the dataset. In Table 10.1, we provide a list of the earliest forgotten samples when we train a BERT model on the second split of SST-2 dataset. The results suggest that SSFT is able to identify mislabeled samples.

Sentences in SST-2 dataset with smallest forgetting time	Label
The director explores all three sides of his story with a sensitivity and an inquisitiveness reminiscent of Truffaut	Neg
Beneath the film's obvious determination to shock at any cost lies considerable skill and determination , backed by sheer nerve	Neg
This is a fragmented film, once a good idea that was followed by the bad idea to turn it into a movie	Pos
The holiday message of the 37-minute Santa vs. the Snowman leaves a lot to be desired.	Pos
Epps has neither the charisma nor the natural affability that has made Tucker a star	Pos
The bottom line is the piece works brilliantly	Neg
Alternative medicine obviously has its merits ... but Ayurveda does the field no favors	Pos
What could have easily become a cold, calculated exercise in postmodern pastiche winds up a powerful and deeply moving example of melodramatic moviemaking	Neg
Lacks depth	Pos
Certain to be distasteful to children and adults alike , Eight Crazy Nights is a total misfire	Pos

Table 10.1: First-split sentences that were forgotten by the 3rd epoch of second-split training of a BERT-base model on the SST-2 dataset. Notice that all of these forgotten examples are mislabeled.

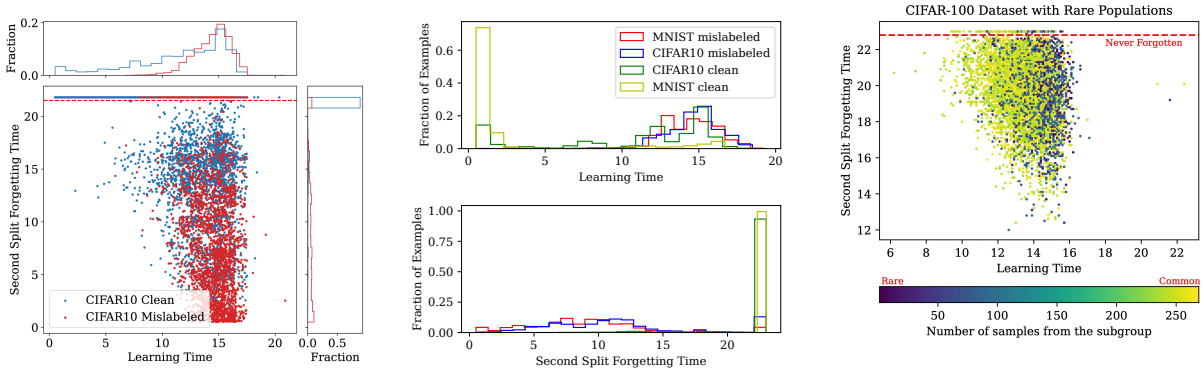
10.4.3 Ablation Experiments

We design specific experimental setups to capture the three notions of hardness as defined in Section 10.3.

Mislabeled Examples We sample 10% datapoints from both the first and second split of the CIFAR-10 dataset, and randomly change their label to an incorrect label. Figure 10.3a shows the learning-forgetting spectrum for the dataset. In the adjoining density histograms, note that a large fraction of the mislabeled and correctly labeled examples are learned at the same time. However, during second-split training, the mislabeled examples are forgotten quickly whereas a large fraction of the clean examples are never forgotten, allowing SSFT to succeed in distinguishing mislabeled samples.

Complex Examples We generate a joint dataset that contains the union of both MNIST and CIFAR-10 examples. This is motivated by work in simplicity bias [Shah et al., 2020] that argue that neural networks learn simpler features first. We also add 10% labeled noise to each of the datasets in the union to understand the learning and forgetting time relationship of a sample that is complex or mislabeled together. In Figure 10.3b, we show the FSLT and SSFT for MNIST and CIFAR-10 samples. We note that a high fraction of the CIFAR-10 (complex) samples learn at the same speed as the mislabeled samples. However, when looking at the SSFT, we are able to draw a strong separation between the mislabeled samples and complex samples. This indicates that the complexity of a sample has low correlation with its tendency to be forgotten once learnt, but a high correlation with being learned slowly.

Rare Examples The CIFAR-100 [Krizhevsky et al., 2009] dataset is a 100-class classification task. The dataset contains 20 superclasses, each containing 5 subclasses. We create a 20-class classification dataset with long tails simulated through the 5 sub-classes within each superclass. More specifically, the number of examples in each subgroup for a given



(a) Mislabeled Examples **(b)** Complex Examples **(c)** Rare Examples

Figure 10.3: Ablation experiments to distinguish the learning and forgetting dynamics for specific types of hard examples. (a) Mislabeled samples may be learned as slowly as a high fraction of typical samples, but they are forgotten much faster. (b) FSLT distinguishes complex (CIFAR10 clean) and simple (MNIST clean) examples, but SSFT does not. On the contrary, FSLT can not distinguish clean and mislabeled examples of CIFAR10, while the SSFT can. (c) FSLT is able to distinguish examples based on the sub-group frequency, however, SSFT has a low correlation with the sub-group frequency.

superclass is given by {500, 250, 125, 64, 32} respectively (exponentially decaying with a factor of 2). This is done to simulate the hypothesis of dataset subgroups following a Zipf distribution [Zipf, 2013] as argued for by Feldman [2020a]. This dataset is further divided into two equal splits to analyze the learning-forgetting dynamics. In order to remove any other effects of example hardness (either within a subgroup, or among subgroups), we randomize both the chosen subset of examples and the ordering of the majority and minority groups between each superclass, by training the model on 20 such random splits and aggregating learning and forgetting statistics over these runs. In Figure 10.3c, we show a scatter plot for the FSLT and SSFT, colored by the frequency of the group a particular example belongs to. We observe that FSLT strongly correlates with the size of the subgroup, whereas the SSFT has a very low correlation with the rareness of a sample.

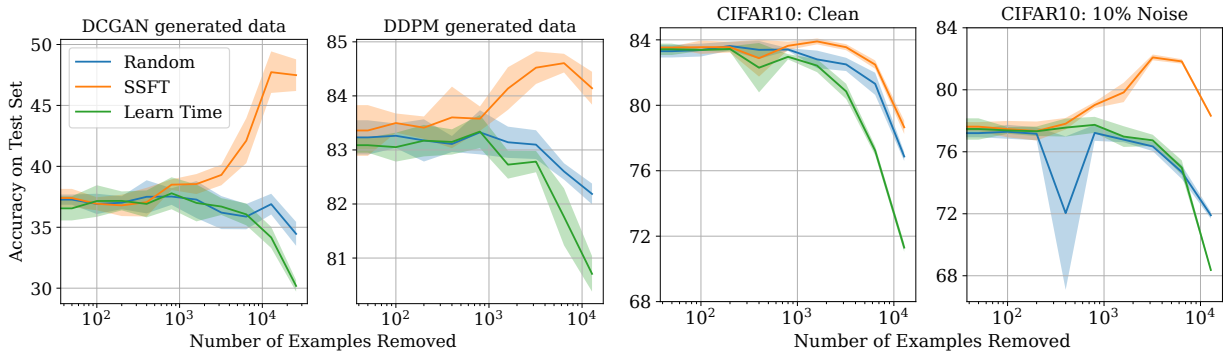
We provide further ablations to show that FSLT is able to identify hard and rare examples, but SSFT shows nearly no discriminative power at finding the two in Appendix E.

10.4.4 Dataset Cleansing

Identifying Label Noise We present AUC scores for detection of label noise via various popular methods in example difficulty literature, across various datasets in Table 10.2. We note that (i) cumulative predictions over the course of training help stabilize both the learning time and forgetting time metrics; (ii) for simple datasets such as MNIST with few ambiguous images, all of the baseline methods have very high AUC (greater than 0.99) in

Method →	t	$\mathbf{acc}_l$	τ (Ours)	$\mathbf{acc}_f$ (Ours)	$\mathbf{conf}_l$	$\mathbf{n}_f$	Joint (Ours)
Imagenette	0.834	0.912	0.931	0.941	0.786	0.781	**0.957**
CIFAR10	0.740	0.900	0.938	0.941	0.947	0.580	**0.958**
MNIST	0.973	**0.998**	0.997	**0.998**	0.965	0.377	**0.998**
CIFAR100	0.700	0.899	0.865	0.885	0.860	0.300	**0.926**
EMNIST	0.987	0.990	0.987	0.989	0.984	0.386	**0.997**

Table 10.2: AUC for identification of label noise using various metrics for example hardness across different datasets. Across all datasets, our τ metric outperforms alternative baselines. We introduce $\mathbf{acc}_f$ as the cumulative accuracy on the second-split training, inspired by previous work that suggests using cumulative accuracies helps make first-split learning time more stable [Jiang et al., 2020]. All other notations are described in Section 10.3. In the case of the Joint method, we select new prediction ranks based on the combined learning and forgetting ranks, further improving over the τ metric alone.



(a) Synthetically generated CIFAR10-like data **(b)** CIFAR10 with and without label noise

Figure 10.4: Accuracy on CIFAR-10 test set after removing a varying number of samples from the training set based on (i) random choice, (ii) examples with the lowest SSFT, and (ii) the highest FSLT. Removing examples based on SSFT helps improve the generalization on the original test set.

finding noisy inputs. However, in datasets such as CIFAR-10 and Imagenette, we find that second-split forgetting metrics do better than first-split training metrics. Finally, we also compare the use of both forgetting and learning time to find noisy samples, and we find a small improvement in the results of just using the forgetting time. While we do not make explicit comparisons with other state of art methods dedicated to finding label noise, our results suggest that augmenting second split forgetting time information may help improve their results. As also observed in recent work [Jiang et al., 2020], we find that the number of forgetting events ($\mathbf{n}_f$) [Toneva et al., 2018] is an unreliable indicator of mislabeled samples. We hypothesize that this is because of the fact that mislabeled examples may often be (first) learnt very late, hence their count of total forgetting events is also low.

Cleaning synthetically generated datasets Generative models are capable of mimicking

the distribution of a given dataset. We generate synthetic datasets of CIFAR10-like samples using (i) DDPM (denoising diffusion model [Ho et al., 2020]); and (ii) DCGAN (Deep Convolutional GAN [Radford et al., 2015]). In both cases, we assign pseudo-labels using the BiT model [Kolesnikov et al., 2020] as in prior work [Nakkiran et al., 2021]. We collect a sample of 50,000 training examples and record the generalization performance on CIFAR-10 as we remove 'hard' samples, as evaluated by various metrics. In Figure 10.4, we can see that removing the most easily forgotten examples can benefit by up to 10% generalization accuracy on the clean test set of CIFAR-10. In case of the synthetic data generated using DDPM, the gains in generalization performance are under 2%. We hypothesize that this is because the samples generated by DDPM are more representative of the typical distribution of CIFAR-10 than those generated by DCGAN.

Note: The ability to train on a second split allows SSFT the *unique* opportunity to train on a clean split of CIFAR-10 in order to assess the alignment of the synthetic samples with the oracle samples. As a result, the SSFT is much more effective in filtering out ambiguous first-split synthetic examples.

10.4.5 Evaluating Example Utility

Recent works [Toneva et al., 2018, Feldman and Zhang, 2020] have argued for removing a large fraction of the less memorized examples, and keeping the memorized ones. We will analyze the change in model generalization upon removing varying sizes of examples from the training set, as ranked by lowest SSFT and highest FSLT (Figure 10.4). In the presence of noisy examples, removing samples based on the SSFT helps improve generalization, whereas FSLT does not do much better than random. We draw the following inferences:

FSLT finds important samples As we remove more samples from the dataset, the accuracy of the model trained after samples are removed based on the highest FSLT is significantly lower than random guessing. This suggests that the utility of these samples is higher than random samples. Put in line with the hypothesis of memorization of rare example as proposed in [Feldman, 2020a], we see that empirically, the examples that are slow to learn are important for the model's test set generalization.

SSFT removes pathological samples On the contrary, removing examples based on the SSFT helps improve model generalization (especially when there is label noise). Even in the setting when there is no label noise, in contrast to FSLT, we find that removing examples that were easily forgotten has a lower negative impact on the model's generalization as opposed to removing random samples. This suggests that the examples that are forgotten in the early epochs of second-split training hurt a model's generalization, and may not be

characteristic samples of their particular class.

Practitioner's view From the AUC numbers in Table 10.2, it may appear that removing examples via learning-based metrics such as learning time and cumulative learning accuracy also provides a high rate of removal of noisy samples. However, when we observe the example utility graphs in Figure 10.4, we draw the inference that the examples that are learned late, are often important examples (such as rare memorized examples). However, even when SSFT fails to capture the correct noisy examples, it still removes unimportant samples and does not hurt generalization. Similar graphs for other metrics described in Table 10.2 can be found in Appendix E.

10.4.6 Characterizing Potential Failure Modes

Figure 10.5: By inspecting the earliest-forgotten examples, we can gain insights into potential failure modes. This model quickly forgets planes with green backgrounds and horses on blue backgrounds.

Recent works have attempted to train classifiers on datasets that contain spurious features [Sagawa et al., 2019, Idrissi et al., 2021] (example Waterbirds, CelebA [Liu et al., 2015] dataset). However, a fundamental challenge is to first identify the spurious correlation that the classifier may be relying on. Only then can recent methods be trained to remove the reliance on spurious patterns. We train a ResNet-9 model to classify CIFAR-10 images of horses and airplanes. In Figure 10.5, we observe that the model forgets planes with green backgrounds and horses with blue backgrounds. This suggests that the model relied on the background as a spurious feature. By analyzing the forgotten examples we can further investigate the examples that the classifier fails to generalize to.

Stability of SSFT We note that SSFT is stable across multiple seeds (Pearson correlation of 0.81), and across architectures (Pearson correlation of 0.63). While the overall correlation for samples ranked by SSFT may be low across architectures, the top-ranked examples have a high correlation (0.85), suggesting the most forgotten examples are consistent across architectures. In contrast, FSLT has a Pearson correlation of 0.52 across seeds. Most interestingly, the learning time metric is brittle to the choice of hyperparameters. As shown by Jiang et al. [2020], when using Adam optimizer, examples of different hardness get learned together. In our experiments, we observe the same phenomenon during learning, however, SSFT is robust to the choice of the optimizer. Detailed results in Appendix E.

Limitations One limitation of the proposed metric is that it is brittle to the choice of the learning rate for the second-split training. If we use a very small learning rate, then overparametrized deep models are capable of learning the new dataset without forgetting

examples from the first split. Alternately, if we use a very large learning rate, the model may diverge and undergo catastrophic forgetting. However, under 'reasonable' choices of learning rate (like that for first-split training), we find SSFT is robust. We provide a detailed anaylsis of the same in Appendix E.

10.5 Conclusion

While many prior works investigate training time dynamics to characterize the hardness of examples, we enrich this literature with a complementary lens focused on the second-split forgetting time. We demonstrate the potential of SSFT to distinguish among rare, mislabeled, and complex examples; and also show the differences in the example properties captured by first-split and second-split metrics.

Our work opens new lines of inquiry in future work that can utilize the separation of hard examples. First, we expect state of art methods in label noise identification to benefit by augmenting our approach. Further, we believe our ablations showing that complex, noisy, and mislabeled samples may all be learned slowly inspire future work that can unite different takes on the memorization-generalization research—early learning, simplicity bias, and singleton memorization.

Chapter 11

T-MARS

Based on Pratyush Maini, Sachin Goyal, Zachary C. Lipton, J. Zico Kolter, Aditi Raghunathan. ICLR 2024

Abstract

We study representation learning without over-reliance on textual shortcuts, introducing T-MARS to improve visual representations by circumventing text feature learning.

11.1 Introduction

Web-scale training demands not only ample data but also *effective* selection: given heterogeneous quality and finite compute, which examples best advance downstream performance? This chapter studies data selection through the lens of training dynamics, introducing T-MARS as a practical method for ranking and curating training data. The approach leverages signals exposed during the learning process—e.g., which examples are learned quickly, persist under distribution shifts, or induce instability—to prioritize informative data and de-emphasize misleading or redundant samples.

We detail the methodology, empirical behavior across datasets, and ablations that clarify when training-dynamics–guided selection helps most. The results support a simple takeaway: selection policies that reflect how models actually learn can yield better data efficiency than static heuristics, especially when high-quality data is scarce or repeated.

11.2 Method

Our pilot study in Sec. 12.3 revealed that a significant portion of the dataset consists of images for which text is the sole feature associated with the caption. Intuitively, these images encourage the model to solve optical character recognition in order to align the image and caption representations. Considering our downstream goal of learning better visual representations, it is natural to filter out such images. However, simply removing

images that contain text matching the caption, is not be optimal due to the presence of images with both visual and text features as seen in our pilot study.

Task: Consider a pretraining image-caption dataset $\mathcal{S} = \{(\mathbf{x}, \mathbf{t})\}^n$, used to train CLIP [Radford et al., 2021a] style models using contrastive learning. Given a fixed computation budget (number of training iterations), our goal is to find a subset of the dataset $\hat{\mathcal{S}} \subset \mathcal{S}$, such that models trained on $\hat{\mathcal{S}}$ have higher zero-shot accuracy on downstream tasks (such as image classification) than those trained on $\mathcal{S}$.

CLIP similarity score: Given image-caption pair $(\mathbf{x}, \mathbf{t})$, the CLIP score refers to the cosine similarity between the embeddings of the image and the caption, i.e., $f_{\text{img}}(\mathbf{x})^\top f_{\text{text}}(\mathbf{t}) / \| f_{\text{img}}(\mathbf{x}) \|_2 \| f_{\text{text}}(\mathbf{t}) \|_2$.

11.2.1 T-MARS: Text-Masking and Re-Scoring

Based on the above hypothesis, we propose a simple and scalable approach, T-MARS, which focuses on evaluating the similarity of *only the visual features* in an image with its corresponding caption. T-MARSfirst masks out the text in the image and then calculates the similarity score between the masked image and the caption using a pre-trained CLIP model. By filtering out images with low masked similarity scores, we retain only those samples where visual features correlate with text.

1. *Text Detection*: We apply a text detection algorithm (FAST [Chen et al., 2021]) that identifies the bounding boxes of text regions in the image (Figure 11.4). Notably, text detection focuses on localizing text positions in the image rather than recognizing or reading the text itself. This key distinction allows our approach to be an order of magnitude more scalable compared to text recognition-based filtering methods [Radenovic et al., 2023].
2. *Text Masking*: Mask the text regions by replacing it with average color of the surrounding pixels.
3. *Re-Scoring and Filtering*: Using a pre-trained CLIP model, we calculate the cosine similarity between the masked image and the original caption. Finally, we simply filter out 50 percent of the datapoints that have the lowest similarity scores between the masked image and the caption. One can also choose to filter the datapoints based on a threshold score on the cosine similarity between the masked image and original caption.

We filter out 50% of the pool and use a simple inpainting technique of neigbouring pixel colors to simplify design choices and they indeed serve us well. Note that we use the corresponding original images for training on the filtered subset, and not the masked images themselves. Algorithm Box 13.7.1 details T-MARS.

We first highlight the empirical effectiveness of T-MARSin Section 11.3.3. Later, in Section 11.4, we (1) show that T-MARSindeed works as intended, filtering out the images

with only text features, while retaining those with both visual and text features, and (2) verify that all inputs with visual features have high positive utility and must be retained. On the other hand, images with only text features hurt as much as mislabeled ones.

11.2.2 Contributed Baselines

Here we briefly describe two baseline filtering approaches, which we propose by drawing insights from hard example mining literature in the supervised training setup (a more detailed description can be found in Appendix **??**). As we show later in Sec. 11.3.3, our proposed baselines themselves outperform the exisiting data curation approaches for training visual language models.

C-SSFT Maini et al. [2022] proposed the SSFT (Second-Split Forgetting Time) to identify mislabeled examples in a dataset by fine-tuning a converged model on validation data, and observing which examples change their predicted label the earliest. Given the absence of a converged model in webscale learning, we use a pre-trained model from OpenCLIP [Ilharco et al., 2021] and finetune for $n = 5$ epochs on the Conceptual-Captions dataset with a learning rate of $1e^{-5}$. We then calculate the average cosine similarity for all examples during the fine-tuning (forgetting) phase and rank examples based on the average similarity score, retaining only the highest-scoring ones.

C-RHO Mindermann et al. [2022] proposed RHO loss to prioritize the training on examples that are worth learning, but not yet learned. In classification, at every epoch, the method calculates the difference between the model's training loss on a given data point and a validation model's loss on it. Examples with low validation loss, but high training loss are prioritized. We propose RHOadapted for web-scale. (1) Rather than using the training loss, we utilize the model's image-caption similarity score; (2) We train our model for one epoch on the entire dataset to calculate the training loss and use a model trained on CC3M dataset as the validation model. Then, we calculate the difference in training and validation similarity scores to rank examples and only keep the top 50% of examples for training.

11.2.3 Existing Baselines

LAION filtering The initial curation of the LAION-400M [Schuhmann et al., 2021] was based on filtering common-crawl samples with a CLIP similarity lower than 0.281 (using OpenAI's CLIP ViT-B/32). Samples with non-English captions are also filtered out.

CLIP Score We also investigate the use of stronger CLIP score thresholding by retaining image-caption pairs with high similarity to further reduce the size of the training pool by 50%. This would mean training multiple epochs on high CLIP-scored data, as opposed to

a single epoch on all the data.

Text Match Radenovic et al. [2023] proposed removing all the images that contain text overlapping with the caption (5 continuous characters) to ensure that the model only focuses on visual features in the dataset. We skip the caption complexity and caption action filtering part, since it is shown to have a negative impact on accuracy in the original paper. Importantly, note that Text Match is 10× more costly than just text masking, and the quality of text recognition in web images is so low that state-of-art recognition algorithms are unable to identify all text in the image correctly. On the other hand, text masking used in our work only requires detection, which is fast and accurate (Appendix 12.3).

11.3 Experiments

We evaluate various baselines (including those laid by this work) as well as our proposed approach T-MARSacross 7 different data pools ranging from 2 million to 128 million. Our results showcase a linear scaling trend in the zero-shot accuracy gains over no data curation, highlighting the importance of incorporating data curation in practice as the data and compute are scaled.

11.3.1 Data Pools and Training Configuration

We first experiment on six different data pools ranging from 2M to 64M samples chosen from the LAION-400M dataset. Note that the compute budget (total training samples seen i.e. epochs × number of batches × batch size) is kept the same as the pool size. For example, for a 32M pool size, the total samples which can be seen during training is kept at 32M (i.e. 1 epoch over the whole dataset). In cases where filtering methods retain a smaller subset (say 16M samples) of the data pool, they get the advantage of running more iterations (2 epochs over 16M subset i.e. total 32M samples seen) over the chosen subset. Finally, we also experiment on the 12.8M (small scale) and 128M (medium scale) data pool of the recently released DataComp. We use the implementation of the Datacomp library to standardize the training process. We train both ResNet 50 and ViT-B-32 models with a batch size of 1024, using cosine learning rate with 200 steps of warmup at $5e^{-4}$. We use AdamW as the optimizer for training. All the experiments were performed on NVIDIA A6000 GPUs.

11.3.2 Evaluation Datasets

We extensively evaluate zero-shot accuracies on a suite of benchmarks considered in prior work [Radford et al., 2021a, Wortsman et al., 2021]: (a) ImageNet: a 1000-class image

			ResNet-50				ViT-B-32			
Scale	Filtering	Dataset size	ImageNet	ImageNet dist. shifts	VTAB	Retrieval	ImageNet	ImageNet dist. shifts	VTAB	Retrieval
16M	LAION	100%	16.63	15.04	24.20	16.79	09.39	08.46	19.83	12.58
	CLIP Score (@ 50%)	50.0%	15.58	14.28	23.67	16.28	09.02	08.42	20.13	12.60
	Text-Match	86.4%	17.83	15.83	24.63	17.11	10.16	08.89	20.63	12.84
	SSFT	90.0%	17.49	15.61	24.90	17.31	10.10	08.94	19.67	13.26
	RHO	50.0%	19.46	17.39	26.45	18.60	10.87	09.34	21.22	13.93
	T-MARS	50.0%	20.25	17.71	26.50	18.45	12.09	10.35	22.64	14.15
	T-MARS∩ SSFT	45.2%	20.81	18.28	26.49	18.96	12.56	10.60	21.96	14.36
	T-MARS∩ RHO	27.5%	**21.63**	**18.62**	**26.70**	**19.53**	**12.61**	**10.94**	**23.48**	**14.58**
32M	LAION	100%	21.90	18.90	27.30	20.18	14.98	12.38	23.21	16.03
	CLIP Score (@ 50%)	50.0%	20.84	18.79	25.71	19.54	14.69	12.86	22.81	15.32
	Text-Match	86.4%	23.80	20.70	28.74	21.41	15.96	13.26	24.45	16.44
	SSFT	90.0%	22.87	19.85	26.10	21.00	15.55	13.34	22.95	16.40
	RHO	50.0%	25.44	21.81	27.65	22.61	16.76	13.98	25.60	17.48
	T-MARS	50.0%	26.73	22.79	29.88	22.62	18.75	15.30	26.71	16.82
	T-MARS∩ SSFT	45.2%	26.89	22.83	28.81	**22.99**	**19.18**	**15.86**	**27.13**	17.82
	T-MARS∩ RHO	27.5%	**27.20**	**23.30**	**30.30**	22.77	19.15	**15.86**	26.93	**18.04**
64M	LAION	100%	26.34	23.24	29.09	23.91	20.37	17.97	27.85	18.83
	CLIP Score (@ 50%)	50.0%	25.66	22.83	29.05	23.36	20.07	17.27	27.55	18.33
	Text-Match	86.4%	29.11	24.94	30.35	25.75	23.11	19.04	28.82	19.37
	SSFT	90.0%	28.15	24.13	29.73	25.58	21.80	18.20	27.69	19.54
	RHO	50.0%	28.66	24.83	30.13	19.79	23.27	19.23	27.94	21.10
	T-MARS	50.0%	32.47	27.52	33.05	24.99	**25.78**	**21.05**	**31.69**	20.52
	T-MARS∩ SSFT	45.2%	**32.77**	**27.68**	**33.13**	**26.35**	25.63	21.01	30.02	**21.27**
	T-MARS∩ RHO	27.5%	32.63	27.23	32.77	25.57	25.62	20.73	31.57	20.63

Table 11.1: Zero-shot accuracies for models trained on filtered subsets of the original LAION dataset when evaluated on a suite of 23 benchmark datasets (Sec. 11.3.2). Rows in 'orange' depict previous baselines (Sec. 11.2.3), those in 'white' depict our contributed baselines (Sec. 11.2.2), and those in 'green' depict our state-of-the-art method T-MARS(Sec. 11.2). ∩ denotes the intersection between two filtering strategies.

	small (12.8M)						medium (128M)					
Filtering	Dataset size	ImageNet	ImageNet dist. shifts	VTAB	Retrieval	Avg.	Dataset size	ImageNet	ImageNet dist. shifts	VTAB	Retrieval	Avg.
No filtering	12.8M	02.5	03.3	14.5	11.4	13.2	128M	17.6	15.2	25.9	25.8	25.8
Basic Filtering	3.0M	03.0	04.0	14.9	11.8	14.2	30M	22.6	19.3	28.4	28.5	28.5
LAION filtering	1.3M	03.1	04.0	13.6	09.2	13.3	13M	23.0	19.8	30.7	29.2	29.2
CLIP score (L/14 30%)	3.8M	05.1	05.5	19.0	11.9	16.4	38M	27.3	23.0	33.8	32.8	32.8
Text-Match	3.5M	05.7	06.2	18.9	12.0	17.3	34M	29.4	24.7	34.4	26.0	34.3
T-MARS	2.5M	06.4	**06.7**	**20.1**	**13.4**	17.9	25M	33.0	27.0	36.3	**29.4**	36.1
T-MARS∩ RHO	1.5M	05.6	05.9	17.8	11.5	17.7	15M	30.3	24.9	34.9	25.3	35.7
T-MARS∩ SSFT	2.3M	**06.5**	**06.7**	19.4	13.1	18.0	23M	**33.8**	**27.4**	**37.1**	28.5	36.2

Table 11.2: Zero-shot accuracies for various filtering strategies on the `small` and `medium` pools of the DataComp benchmark. ∩ denotes the intersection between two filtering strategies. T-MARSoutperforms the state-of-art on DataComp by a margin of 5% on the medium scale (ImageNet).

classification challenge [Russakovsky et al., 2015]; (b) ImageNet-OOD: Six associated imagenet distribution shifts—ImageNetV2 [Recht et al., 2019], ImageNet-R [Hendrycks et al., 2020], ImageNet-A [Hendrycks et al., 2019], ImageNet-Sketch [Wang et al., 2019], ImageNet-O [Hendrycks et al., 2019], and ObjectNet [Barbu et al., 2019]; (c) VTAB: 12 datasets from the Visual Task Adaptation Benchmark [Zhai et al., 2020], including Caltech101, CIFAR100, DTD, Flowers102, Pets, SVHN, Resisc45, EuroSAT, Patch Camelyon, Clevr Counts, Clevr Distance, KITTI and Sun397; and (d) Retrieval: 3 retrieval tasks of MSCOCO [Chen et al., 2015], Flickr [Young et al., 2014] and WinoGAViL [Bitton et al., 2022].

11.3.3 Results

T-MARSgives impressive gains in accuracy over 23 downstream tasks of various types. Table 11.1 compares zeroshot accuracies of various data curation strategies under pool sizes of 16M, 32M and 64M. First, note that T-MARSconsistently outperforms the baselines across the data pools. For example, on the 64M subset, T-MARSobserves 6.4% gains on ImageNet zeroshot accuracy over no filtering and 3.7% gains over text matching. Similarly, T-MARSoutperforms text-matching by 2.7% in average accuracy over 6 ImageNet dist. shift datasets and by 2.78% in accuracy over 13 vision tasks of the VTAB benchmark (Table 11.1). Results on 2M, 4M and 8M pool sizes are in Appendix **??**.

Complementary data subsets A very important observation from our work is that the data subsets filtered out by the three approaches proposed in our work have large fractions of exclusive subsets (see column data size). This observation translates into the fact that taking the intersection of data retained by different algorithms (T-MARS, SSFT, RHO) has additive benefits.

Scaling Trends An important consideration when proposing and evaluating any data filtering approach is whether or not the gains observed will continue to stay as the scale of

data or compute grows. We present scaling trends for various techniques in Figure 12.3, 12.3 which show that the gains in the zero-shot accuracy has a near linear slope as the data and compute are scaled exponentially (on the x-axis). This is extremely promising as it suggests that rather than gains saturating, gains offered by our method will grow logarithmically with the scale of the total data pool and compute.

Higher accuracy using half the compute and half the data We observe that selecting a better subset of data can be of higher utility compared to adding new unfiltered samples. For example, T-MARS∩ RHOfiltered subset from the 32M pool gives an Imagenet accuracy of 27.20% at a compute of 32M, which is around 1% more than that of the LAION 64M data pool even at double the compute of 64M. This highlights the importance of incorporating data curation in practice, rather than expending additional compute on new unfiltered samples. In Section 11.4, we show a similar observation—there is higher utility in filtering out bad samples in comparison to adding new samples.

State of Art on DataComp We also evaluate T-MARSon the recently released data filtering benchmark DataComp (Table 11.2). For DataComp, we filter out the samples with a similarity score (between the masked image and original caption) below 0.281. Since the first release of T-MARS, there have been 10+ new methods on the Datacomp benchmark on the 'medium scale'. Notably, T-MARSand T-MARS∩ SSFTare 2 of the top 3 entries on the leaderboard. Other top-performing entries require a mixture of multiple rules to achieve comparable or worse results than T-MARS. On the 'medium' scale, our proposed approach outperforms CLIP filtering by a large margin of 6.5% and text-filtering by 4.4%. Datacomp leaderboard has another track 'bring your own dataset'. In this category, Nguyen et al. [2023] use the synthetic captions from BLIP [Li et al., 2023a] to replace noisy web captions. Despite the expensive process of creating new data, rather than filtering existing subsets, T-MARSperforms comparably on average, and even outperforms it by over 2% on Imagenet. This highlights the importance of filtering out the images without visual features. Based on the aforementioned scaling trends, our results portray an optimistic picture for practitioners with more compute budgets to implement T-MARSat the largest scale.

11.3.4 T-MARSeffectively filters targeted images

Recall the pilot study in Section 12.3. T-MARSfiltered out a total of 250 images (out of 500 datapoints) and indeed works as expected by filtering out 95 of the 103 "text dominated" images, while also successfully retaining 46 out of 96 images that exhibit both visual and text features (in contrast, text match-based filtering retained only 21). CLIP score can be a noisy metric that is not well-calibrated across various images. Consequently, we also

observe that in addition to removing text-dominated images, T-MARSalso filtered out 76 of the 234 images with visual features only, because of their low alignment with the caption. That said, we do note that simply filtering based on CLIP score without masking (CLIP Score row, Table 11.1) performs even worse than no filtering, highlighting the significance of incorporating masking in T-MARS. We discuss further details in Appendix 12.3.

11.4 Limitations and Conclusion

We present a state-of-the-art data filtering approach, Text-Masking and Re-Scoring (T-MARS), for web-scale datasets. T-MARSfilters out examples where text dominates visual features in images, improving visual representation learning. Extensive evaluations on LAION and DataComp benchmark demonstrate that T-MARSoutperforms competitive baselines by accuracy gains as high as 3.3% and 6.5% respectively. Data filtering can potentially introduce bias in the filtered subset. However, our text filtering approach is generic and does not incorporate any sort of group bias. Developing further nuanced metrics to rank the masked images beyond the CLIP score is an interesting direction for future work. While in this work, we create a *static* subset of the original corpus and perform multiple epoch training over the same, future work may benefit by assessing the utility of different data points in a dynamic fashion and refreshing the data pool with samples worth learning.

Acknowledgements

We thank the Datacomp and OpenCLIP teams for their code base that was used for training models in our work. ZL gratefully acknowledges the NSF (FAI 2040929 and IIS2211955), UPMC, Highmark Health, Abridge, Ford Research, Mozilla, the PwC Center, Amazon AI, JP Morgan Chase, the Block Center, the Center for Machine Learning and Health, and the CMU Software Engineering Institute (SEI) via Department of Defense contract FA8702-15-D-0002, for their generous support of ACMI Lab's research. AR gratefully acknowledges support from Open Philanthropy, Google, Applw funding from the Bosch Center for Artificial Intelligence. PM is supported by funding from the DARPA GARD program.

Chapter 12

FADU

Based on Pratyush Maini et al.. Preprint

Abstract

Vision-language models (VLMs) are trained for thousands of GPU hours on carefully curated web datasets. In recent times, data curation has gained prominence with several works developing strategies to retain 'high-quality' subsets of 'raw' scraped data. For instance, the LAION public dataset retained only 10% of the total crawled data. However, these strategies are typically developed agnostic of the available compute for training. In this paper, we first demonstrate that making filtering decisions independent of training compute is often suboptimal—the limited high-quality data rapidly loses its utility when repeated, eventually requiring the inclusion of 'unseen' but 'lower-quality' data. To address this quality-quantity tradeoff (QQT), we introduce neural scaling laws that account for the non-homogeneous nature of web data, an angle ignored in existing literature. Our scaling laws (i) characterize the *differing* 'utility' of various quality subsets of web data; (ii) account for how utility diminishes for a data point at its 'nth' repetition; and (iii) formulate the mutual interaction of various data pools when combined, enabling the estimation of model performance on a combination of multiple data pools without ever jointly training on them. Our key message is that data curation *cannot* be agnostic of the total compute that a model will be trained for. Our scaling laws allow us to curate the best possible pool for achieving top performance on Datacomp at various compute budgets, carving out a pareto-frontier for data curation.

12.1 Introduction

Modern pretraining corpora contain a mix of prototypical, easy-to-learn examples and rare, noisy, or conflicting ones that models tend to memorize and later forget. To allocate limited compute effectively, we would like to quantify the *utility* of data through the lens of learning and forgetting dynamics: which examples contribute durable signal and which mainly induce memorization or brittleness?

This chapter develops a forgetting-aware view of data utility (FADU). By tracking

how examples are learned and subsequently forgotten, we identify subsets that are high-impact versus those that are redundant or harmful. We present methodology, empirical evaluation, and practical guidance for using forgetting signals to curate robust training sets and improve data efficiency.

12.2 Scaling Laws for Data Filtering

12.2.1 Defining Utility

Past works on scaling laws [Kaplan et al., 2020, Jia et al., 2021] estimate the error of a model (at a given parameter count) after training for n samples as: $y = an^b + d$, where $a, d > 0$ and $b < 0$ are constants to be determined empirically, and y is a performance metric such as the loss of the model on a validation set. Intuitively, b factors in in the diminishing gains as more data is seen and also models the utility of the data pool itself, with a lower value indicating higher utility. For instance, Cherti et al. [2023] noted that the b value for OpenAI's filtered dataset was lower than that of the LAION dataset, indicating it had higher utility. Whereas, a is a normalizer and d estimates an irreducible error at the end of training to infinity. Rather than estimating the loss at the end of training for n samples, we can also consider the instantaneous utility of a sample at any point during training. This is given by:

$$\frac{dy}{dn} = a \cdot bn^{b-1} = \frac{y}{n} b. \tag{12.1}$$

This equation shows that the instantaneous utility of a sample is proportional to the current loss and inversely proportional to the number of samples seen so far. This is intuitive as the utility of a sample decreases as more data is seen. Crucially, note the data utility parameter b.

12.2.2 Utility under Repetition

Now, let us add one more complexity to this scaling law from past works. In practice, CLIP style pretraining is done for multiple epochs on the same data [Gadre et al., 2023]. However, there is no clear understanding of how the utility of a sample changes with repetition. We hypothesize that this utility decays exponentially with the number of times the sample is seen. More formally, the utility parameter (b) of a sample seen $k + 1$ times is given by:

$$b_{k+1} = b \cdot \left(\frac{1}{2}\right)^{\frac{k}{\tau}} = b \cdot \delta^k \tag{12.2}$$

where τ is the half-life of the utility parameter. A higher value of τ indicates that the utility of a sample decays slower with repetition. δ more concisely captures the decay in utility with repetition, and is used for simplicity of notation. Then, a closed form expression of the loss of a model after seeing n samples k times each is given by:

$$y_k = a \cdot n_1^{b_1} \prod_{j=2}^{k} \left(\frac{n_j}{n_{j-1}} \right)^{b_j} + d \tag{12.3}$$

where n_j is the number of samples seen at the end of j^{th} epoch of training. The equation is derived in Appendix F and forms the basis of our scaling law.

Summary of Parameters Let us concisely summarize the role of each of the parameters in our scaling laws (Eq. 12.3) in order to develop better intuition about each of them.

1. **Utility Parameter** (b): The change in loss scales with the number of samples seen exponentially based on the value of b. A high quality data bucket will have a lower b value (negative with high magnitude) compared to a worse data bucket.

2. **Half life** (τ): The repetition parameter captures the decay in the utility of repeated data. Intuitively, the half life τ captures the diversity of the data bucket. Data buckets with high diversity will have a higher value of τ, allowing more repetitions of the bucket, as one would desire.

3. **Decay Parameter** (δ): The decay parameter is a parameter directly derived from τ, and not a unique parameter. We use this for simplicity of notation. δ captures the fractional decay in the utility parameter with one epoch of training on that data.

4. **Normalizer** (a): The normalizer aims to capture an intrinsic property of the task allowing us to relate the change in loss with the number of samples seen. This does not change with the bucket. We learn a common value of a that minimizes the loss for all buckets, and treat it as a fixed constant across all buckets.

5. **Irreducible loss** (d): This is a constant parameter added to the loss that can not be reduced further.

12.2.3 The case of heterogeneous web data

Now we are ready to add the final layer of complexity to our scaling laws, that of heterogeneous webdata. A unique challenge in the paradigm of webdata, and critically

missed in the existing works on scaling laws, is the presence of data pools of different quality. As discussed in Section 12.3, webdata can generally be partitioned into multiple subsets (like using clip score), each with it's own respective scaling parameters (like respective data utility parameter b).

Training large scale models then involves jointly training on a mixture of multiple data buckets. This brings us to us central question—*how can we estimate the loss and thus the scaling curves for a mixture of pools effectively*? This ultimately allows to curate the data conditional to any compute, rather than a static curation. One naive way to estimate the error on training on multiple data mixtures would be to use the average error on them. However, this does not factor in the interplay of the two different b values in the exponent of the scaling curve, and how does the repetition parameter (τ) change with increasing data mixtures.

Axiom 1. *Given p data pools $\mathcal{S}_n^1 \ldots \mathcal{S}_n^p$ sampled uniformly at random, with respective utility and repetition parameters given by $(b_1, \tau_1) \ldots (b_p, \tau_p)$, then the new repetition half-life of each of the buckets $\hat{\tau} = p \cdot \tau$. Additionally, the effective utility value for the combined pool b_{eff} at the k^{th} repetition is the weighted mean of the individual utility values. Formally,*

$$b_{eff}^{(k)} = \frac{\sum_i^p b_i \hat{\delta}_i^k}{p}, \tag{12.4}$$

where $\hat{\delta}_i = \left(\frac{1}{2}\right)^{1/\hat{\tau}}$, the new decay parameter per bucket.

We refer the reader to Appendix F for the derivation of the formulae for b_{eff}. We assume that the utility of a sample decays exponentially on being seen multiple times. However, there is one major challenge of contrastive training paradigm, where the effective number of samples in a data pool of size N is N^2. This is because each sample is paired with every other sample in the data pool. In Appendix F we show that $\hat{\tau} = \frac{\hat{N}}{N}\tau$ where $\hat{N}$ is the total number of samples in the data pool and τ is the half-life of the individual pool with N samples.

Finally, one can use the b_{eff} from Theorem 1 in Equation 12.3, to estimate the loss while training on a combination of data pools.

12.2.4 Various other formulations

While deciding the scaling laws for data filtering, we considered various other formulations. This included scaling laws that model the decay in 'effective samples seen' [Muennighoff et al., 2023] rather than effective utility like in this work. In Appendix F, we talk in detail about why an "effective number of samples seen" based formulation cannot model the heterogeneous webdata and other design considerations. Further, we also study various

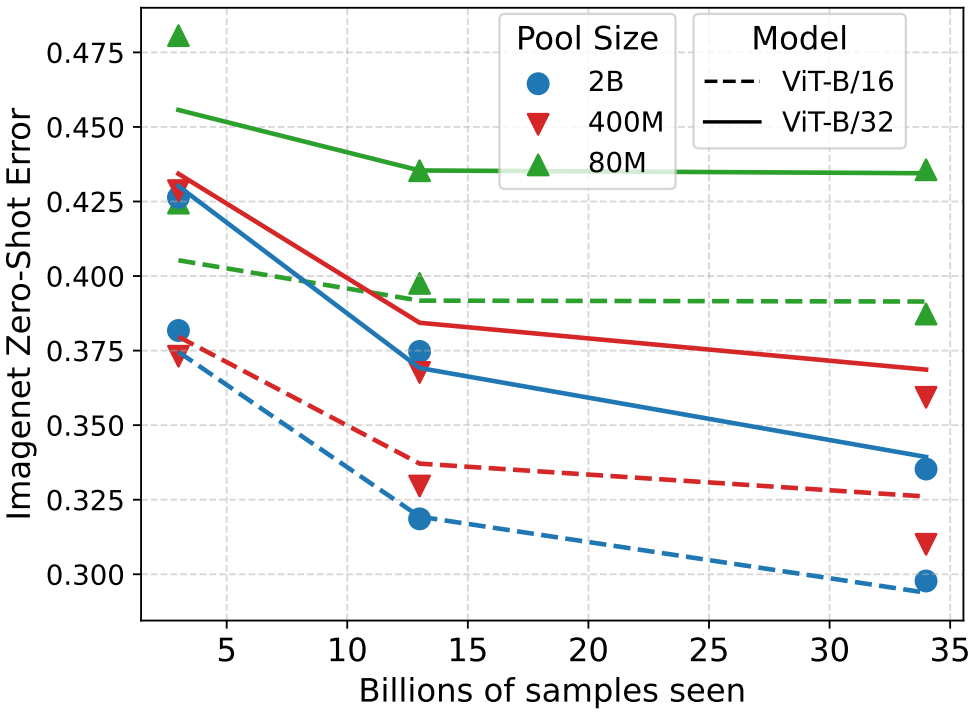


Figure 12.1: Similar to Figure 5, our scaling law accurately predicts the final error for models trained on 2 different architectures, 3 different very large scale pool sizes and 3 different billion scale compute budgets.

choices such as the need for allowing different data buckets to have different 'half lives' (τ), but a unified normalizer (a), and the way we optimized various scaling parameters in Appendix F.

In Sec. 12.3, we first estimate the scaling parameters for individual data pools of various quality. Finally, in Sec. 12.3 we use the scaling parameters of the individual pools to estimate the scaling curves for data combinations.

12.2.5 Scaling the scaling curves

Past work on scaling laws for CLIP models [Cherti et al., 2023] trained tens of models at varying compute scales ranging from 3B to 34B training samples and models spanning different ViT families. While training models at this compute is extremely expensive, we utilize their pretrained models[1]. Cherti et al. [2023] tried to fit scaling laws for this family of models, but the scaling curves showed extremely high errors for models trained on small datasets. We believe this is primarily because they do not account for the impact of

[1]Note that these models were trained by Cherti et al. [2023] much before the recently pointed out issues in LAION.

diminishing utility of repeated data. We use our proposed scaling laws to estimate errors for the models in question. The revised scaling trends are presented in Figure 12.1, which are able to predict the error with a high accuracy. This confirms that our scaling laws hold at massive models trained for 34B data compute, indicating that the diminishing utility of repeated data must indeed be accounted for while predicting model training outcomes.

12.3 Discussion

State of Data Curation Despite recent efforts, the curation and utilization of data remains surprisingly ad-hoc and *hacky*, with very little predictability about the outcomes of a filtering strategy. In particular, all prior filtering approaches (i) propose a metric that ranks examples and filters out data points below a threshold; and (ii) are the thresholds are chosen 'agnostic' of the compute the model is supposed to be trained for. While well-resourced organizations can embark on exhaustive sweeps of 'filtering' parameters, this approach (i) is extremely expensive, especially in the paradigm of web-scale pre-training; and (ii) does not transfer to new training paradigms where one changes the training samples to pool size ratios.

Our scaling laws enable practitioners to precisely assess and quantify the utility of different web data subsets, which is critical given that webdata is heterogeneous. Finally, we show how one can estimate scaling law for a mixture of pool (Theorem 1). This enables a *compute aware* data curation, where one can decide the filtering threshold (which pools to use for training) based on the estimated accuracies using the scaling law for the mixture of pools.

State of Scaling Laws To the best of our knowledge, all scaling laws to date have modeled web data with a singular set of scaling parameters, irrespective of the specific formulation of the scaling law. As we venture into the era of large-scale foundation model training, where data curation is a critical step, our work takes significant steps towards estimating the performance of models over various possible choices of combinations of different data quality pools.

Part V

Synthetic Data and Safety

Synthetic Data and Safety

As high-quality web data plateaus, synthetic data becomes a lever for scaling quality without linearly scaling cost. This Part develops methods to rephrase the web into cleaner, task-aligned corpora; compares source-rephrasing to generator-driven synthesis; and studies failure modes such as hallucinations that can compound under recursive training. We also introduce safety-first pretraining interventions that raise refusal and robustness without sacrificing core capability, accompanied by standardized reporting.

Taken together, these chapters chart a path to scale responsibly: use synthetic data to accelerate and align learning, detect and prune unsafe artifacts, and document safety impacts at pretraining time.

Guiding questions

- Q1: When does rephrasing real data outperform de novo generation, and along which axes (style, diversity, quality)?
- Q2: How do we detect and control hallucinations in diffusion models, especially under prompt or data distribution shifts?
- Q3: Which pretraining-time safety interventions most improve robustness, and how should we evaluate/report them?
- Q4: How do synthetic data strategies interact with compute budgets and downstream evaluation styles?

Chapter 13

Rephrasing the Web: WRAP

Based on Pratyush Maini, Skyler Seto, He Bai, David Grangier, Yizhe Zhang, Navdeep Jaitly. ACL 2024

Abstract

Large language models are trained on massive scrapes of the web, which are often unstructured, noisy, and poorly phrased. Current scaling laws show that learning from such data requires an abundance of both compute and data, which grows with model size. This is increasingly infeasible due to high pretraining costs and the scarcity of high-quality web text. We propose Web Rephrase Augmented Pre-training (WRAP), which uses an instruction-tuned model to paraphrase web documents into targeted styles (e.g., Wikipedia-like or question–answer format) and jointly pre-trains on real and synthetic rephrases. On C4, WRAP speeds up pre-training by 3× at the same compute, improves perplexity by >50

13.1 Introduction

Web text is abundant but noisy, inconsistent in style, and often misaligned with downstream evaluation tasks. Given fixed compute, the question is not only *which* data to select but also whether we can *improve* the data we already have. This chapter studies WRAP (Web Rephrase Augmented Pre-training): pretraining on a mixture of real web text and synthetic rephrasings produced by an instruction-tuned model.

The core idea is simple: transform raw pages into cleaner, task-aligned styles while preserving factual content. We describe prompting strategies for rephrasing, how synthetic data is mixed with real text, and empirical results showing improved sample efficiency and robustness. WRAP illustrates that carefully generated synthetic text can complement curation, yielding more learning per token without expanding the crawl.

13.2 T-MARS: Web Rephrase Augmented Pretraining

Generating synthetic data using an off-the-shelf language model can be both computationally expensive and operationally challenging. Prior approaches to generating synthetic textbook quality data using LLMs [Gunasekar et al., 2023] required (1) a language model that contains sufficient world knowledge to generate articles worth training on, thereby increasing generation cost; (2) a careful selection of prompts that enable generating high quality, and diverse articles that fill any knowledge gaps in the synthetic corpus. This challenge was highlighted in follow-up work of Li et al. [2023d], and has the potential of inadvertently creeping in biases in the language models [Maini, 2023], as opposed to those trained on the natural diversity of the web. As a remedy to the challenge of (i) generation cost, and (ii) data diversity, we propose T-MARS that leverages the natural diversity of articles on the web, allowing us to utilize significantly smaller LLMs (than GPT-3.5) to generate high-quality paraphrases of noisy and unstructured articles on the web.

13.2.1 Rephrasing the Web

It has been observed in past work that up-weighting high-quality data, such as texts from Wikipedia, can be useful to improve language modeling. These terms have generally been very loosely defined and there is only anecdotal evidence of the same [Brown et al., 2020a, Wenzek et al., 2019]. At the same time, web data is deficient of text in question-answering or conversational format, which is a prominent use case of language models. Based on these two insights, we design our rephrasing styles.

Rephrasing Styles In lieu of the anecdotal evidence above, we attempt rephrasing documents on the web in four different styles—(i) Easy (text that even a toddler will understand); (ii) Medium (in high quality English such as that found on Wikipedia); (iii) Hard (in terse and abstruse language); (iv) Q/A (in conversation question-answering format). In order to operationalize rephrasing in these stylistic variations, we appropriately prompt an instruction-tuned model. The rephrased examples of these four styles and the prompts templates used in our work are provided in Appendix H.

Generating Synthetic Data Now, we detail how we utilize an instruction-tuned language model to rephrase texts from web-crawled datasets such as C4 [Raffel et al., 2020] (which we use for all our experiments). In particular, we use a frozen Mistral-7B instruction-tuned model [Jiang et al., 2023a] (see Ablations in Section 13.5 for other models). To generate synthetic data in "medium" style, the Mistral model is prompted using the following

instruction: *"For the following paragraph give me a paraphrase of the same in high-quality English language as in sentences on Wikipedia"*. The prompt was created using iterative human feedback by comparing outputs of 'medium' sized LLMs with those of GPT-4. We use the model output to create a parallel corpus of "high-quality" synthetic data corresponding to the original noisy web data. Each example is chunked into sequences of nearly 300 tokens using the NLTK sentence splitter. This was based on our empirical observation that asking an LLM to rephrase more than 300 tokens often led to a loss of information. They are later coalesced to pre-train with a context length of 1024. Discussions on data quality can be found in Section H.

Combining Real and Synthetic Data Our method of re-phrasing web data naturally incorporates the information diversity found on the internet. However, it does not incorporate the noise in real data. While synthetic data may help LLMs pre-train faster, we also want them to be able to understand noisy web text that may be filled with typos and linguistic errors so that the LLMs do not fail in user-facing situations. In order to incorporate this style diversity in language modeling, we sample real and synthetic data in a 1:1 ratio. Note that this means the same raw data is seen twice, once in its original form, and once as its rephrase.

13.2.2 Implementation Details

Architecture We train decoder-only transformer models [Vaswani et al., 2017] at three different scales, small, medium and XL. The small-scale (128M parameter) model consists of 12 layers, 12 attention heads, and a hidden dimension size of 768. The medium-scale (350M parameter) model consists of 24 layers, 16 attention heads, and a hidden dimension size of 1024. The XL-scale (1.3B parameter) model consists of 24 layers, 16 attention heads, and a hidden dimension size of 2048. We do not use dropout in either model and a maximum sequence length of 1024. The models are trained using NVIDIA's Megatron-LM repository (https://github.com/NVIDIA/Megatron-LM).

Pre-training We train all our XL models for a total of 300k steps with a batch size of one million tokens, unless otherwise specified. We use a maximum learning rate of $3e^{-4}$ for the 128M, and 350M parameter models, and $2e^{-4}$ for the 1.3B parameter model. The minimum learning rate is $1e^{-5}$. We use a weight decay of 0.01, along with a gradient clipping norm of 1.0. We use cosine learning rate scheduler with a warmup for 1% of total steps; and Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

Figure 13.1: T-MARS (C4 + QA-85B) v/s C4: Comparison of perplexity on the Pile for a 1.3B LLM trained for 300B tokens shows that WRAP outperforms models trained on 2x real data.

13.3 Perplexity Evaluation

We evaluate the perplexity of the pre-trained model on the validation set of multiple out-of-distribution datasets. All models are either trained on the C4 dataset [Raffel et al., 2020], or a particular stylistic rephrase of the same. All the evaluations are done on 21 sub-domains of the Pile [Gao et al., 2020]. These subsets are created from the first 10,000 documents from each domain of the Pile dataset. We then evaluate the perplexity of the model on these subsets. Additional evaluation details are provided in Appendix H. It is important to note that we evaluate perplexities on the Pile instead of C4. Training on multiple distributions of text (synthetic and real web) does comes at a marginal cost of less than 1 perplexity on the C4 validation set. To understand our choice of evaluation, and why we observe this perplexity increase, we note that training over the C4 corpus corresponds to minimizing the objective

$$\theta_{c4} = \min_{\theta} \mathbb{E}_{x \sim D_{c4}} \left[\mathcal{L}(\theta; x) \right], \tag{13.1}$$

that attempts to exactly model the C4 web text. In contrast, training over multiple styles corresponds to minimizing the risk over a different distribution,

$$\theta_{T-MARS} = \min_{\theta} \mathbb{E}_{x \sim D_{c4} \cup D_{syn}} \left[\mathcal{L}(\theta; x) \right]. \tag{13.2}$$

Solving for equation 13.2 does not minimize the risk over C4-only, and hence the small drop in performance between θ_{c4} and θ_{T-MARS} on the C4. For meaningfully comparing models trained on the C4 and on its synthetic rephrases, we evaluate their generalization capability on 21 different domains of the Pile [Gao et al., 2020]. Results for each domain are presented in Figure 13.7.1.

Data Complexity In Figure 13.7.1, we show that models trained for fewer tokens (150B) and even smaller 350M models outperform training on the full C4 for 300B tokens indicating faster learning with synthetic rephrases. On some domains such as ArXiv and HackerNews, we observe that training with synthetic data allows reducing the perplexity by nearly 3x of the perplexity of models trained on real data alone. This suggests that in many cases it may not be possible to offset the performance advantage of pre-training on synthetic data by merely training on more real data. On an average of 21 subsets of the Pile, our models improve perplexity by 50% over models trained on real data alone.

Learning Speed We observe that even at the first checkpoint (10B tokens) of T-MARS training, the average perplexity of the LLM on the Pile is lower than that achieved by pre-training on C4 for 15 checkpoints. This suggests a 15x pre-training speed-up. We defer the discussion on learning speed to 'zero-shot' tasks in order to make more meaningful comparisons.

13.4 Zero-shot Tasks

We now evaluate our pre-trained language models on various zero-shot question answering (QA) benchmarks using the LLM Evaluation Harness[1] [Gao et al., 2024c].

13.4.1 Datasets

We evaluate our models on a total of 13 different zero-shot benchmarks to assess their abilities across various natural language tasks like common sense reasoning, language and knowledge understanding and mathematical reasoning.

General Understanding The General Understanding category comprises datasets testing broader cognitive skills and language comprehension. **ARC Easy (ARC-E)** [Clark et al., 2018c] is the less challenging counterpart of ARC-C, featuring questions that require basic reasoning skills. **BoolQ** [Clark et al., 2019] includes boolean questions that focus on reading comprehension and general language understanding. **Winogrande (Wino.)** [Sakaguchi et al., 2021] challenges models with common sense reasoning in language, particularly in pronoun disambiguation. **PIQA** [Bisk et al., 2020a] assesses understanding of physical processes, an essential part of practical common sense. **HellaSwag** [Zellers et al., 2019] tests the ability to complete scenarios coherently, demanding both language understanding and common sense. **TruthfulQA** [Lin et al., 2021] is centered on generating truthful, accurate answers, thus testing the model's factual correctness. **OpenBookQA (OBQA)** [Mihaylov et al., 2018a] evaluates the understanding of a broad range of facts and concepts. Finally, **LogiQA-2** [Liu et al., 2023a] assesses the model's capacity to comprehend logical principles.

Specialized Knowledge In the Specialized Knowledge category, we include datasets that demand expertise in specific domains. The **ARC Challenge (ARC-C)** [Clark et al., 2018c] contains challenging science exam questions from grades 3 to 9, demanding advanced knowledge. **SciQ** [Johannes Welbl, 2017] provides science exam questions to test model understanding and reasoning in the scientific domain. **PubMedQA** [Jin et al., 2019]

[1]We use git commit - `89618bf8` for consistency across all experiments with a batch size of 32.

focuses on comprehension in biomedical literature. **MathQA** [Amini et al., 2019] tests mathematical problem-solving, requiring both numerical comprehension and reasoning. Lastly, **MMLU** [Hendrycks et al., 2021d] spans multiple domains, from professional to academic, testing the model on specialized subjects.

13.4.2 Results

We compare the performance of a model trained on a mixture of real and synthetic data with models trained on various splits of real data. In all our experiments, we use the C4 [Raffel et al., 2020] dataset for rephrasing and producing splits of synthetic data. We use the abbreviation 'Real Tok.' to denote the number of tokens of web data available for pre-training. In the 'Synthetic + Real' experiments, we augment the same number of synthetic rephrases. We choose 'Real Tokens' as the metric of comparison because we can potentially rephrase the same document multiple times, implying that the total corpus size is not meaningful, and corpus 'knowledge' is the actual currency of interest.

Baselines Methods We pre-train LLMs of (i) Half of C4, and the (ii) Full C4 corresponding to approximately 85 Billion and 170 Billion real tokens respectively [Raffel et al., 2020]. We also pre-train our own models on (iii) 160 Billion and (iv) 320 Billion tokens of the RefinedWeb Dataset [Penedo et al., 2023]. We also compare with the (iv) Pythia-1.4B model that was trained on the Pile [Gao et al., 2020]. This dataset is no longer publicly available, hence we utilize a pre-trained model. Finally, we also compare with the recent (v) TinyLlama model [Zhang et al., 2024c] that was trained for 3 epochs on 1 Trillion tokens from SlimPajama [Shen et al., 2023] and StarCoder [Li et al., 2023b].

General Improvements Across all tasks in Table **??**, we observe that models trained on synthetic data combined with the C4 dataset (Synthetic+C4) exhibit an overall higher average performance of 52.3% as compared to those trained solely on the real C4 dataset with a 85B token split, which scored an average of 50.2%. This shows that the inclusion of synthetic data can enhance the general understanding capabilities of NLP models. Moreover, even the TinyLLama model trained for 10x compute and data, performs comparably to models trained on real data only. This suggests that the gains from filtering out, or adding more real data are low. As opposed to this, T-MARS shows that pre-training on even small amounts of synthetic data can contribute to large performance gains.

Specialized Knowledge Tasks The key message from the results in Table **??** is that synthetic data can not impart 'new knowledge'. It can only help pre-train faster, which was

Figure 13.2: Importance of High Quality Paraphraser: Perplexity across all the Pile domains for T-MARS on data generated by different LLMs. Results show that even small models like Qwen-1.8B can generate paraphrases of high quality. Though, a low quality rephraser like our fine-tuned T5-base model leads to worse PPL.

also the premise of our work. In particular, we note several key findings:

1. Pre-training on larger datasets helps improve performance, by presumably exposing the LLM to more "knowledge". For instance, the Pythia (300B) model achieves an average score of 43.4%, outperforming the smaller C4 (85B) dataset's score of 42.1%.

2. Despite the advantages of a larger dataset, the improvements saturate. For example, RefinedWeb (320B) model outperforms the RefinedWeb (160B) model by only 0.6%. Similary, the TinyLlama model (1T tokens) performs comparably to the T-MARS model, which only had 85B tokens of raw web data.

Specific Improvements We see maximum improvement in the TruthfulQA dataset, with the Synthetic (85B) model scoring 44.0%, which is significantly higher than any other model's performance on this dataset. This is potentially because instruction-tuned LLMs already correct potential misconceptions while rephrasing the text. Interestingly, we notice how adding real data to the synthetic model (Synthetic+C4) reduces the performance on TruthfulQA by 4%, down to 40.5%, indicating a potential dilution of the benefits gained from synthetic data when combined with real data. Other datasets such as HellaSwag, and BoolQ, for which C4 trained models do well, continue to show the benefits of incorporating combinations of C4 and synthetic rephrases.

13.5 Analysis and Ablations

We further ask the following Research Questions (RQs) to investigate in a finer granularity how to enhance performance optimally. We discuss additional data and method ablations in Appendix H.

RQ1: How important is to have a high-quality re-phraser? We use data from four distinct re-phrasing models (T5-base [Raffel et al., 2020], Qwen-1.8B-chat [Bai et al., 2023a], Mistral-7B-chat [Jiang et al., 2023a], and Vicuna-13B-chat-v1.3 [Chiang et al., 2023]) and train a 345M model for 30B tokens. We generate data from all models using the same prompt. In case of the T5-base model, we finetune the model for 1 epoch on re-phrase pairs from the Vicuna-13b-chat model. We find that pre-training on data generated by smaller

Figure 13.3: Importance of Real Data: Comparing perplexity on the Pile when pre-training on C4 with synthetic data vs. synthetic data only. Models are 1.3B parameters trained for a total of 150B tokens on a real data subset containing 35 Billion tokens of the C4.

re-phrase models like Qwen-1.8B and Mistral-7B achieve lower perplexity than Vicuna 13B (Figure 13.7.1). At the same time, our fine-tuned T5-base model performs significantly worse than the rest. Even then, all rephrase models reduce perplexity over only real C4 data. It remains an open question to test the limits of how small can we train a paraphrase model to generate high quality synthetic data to further scale the applicability of T-MARS.

RQ2: How important is it to have real C4 data? Our findings in Tables **??** indicate that synthetic data using the QA prompt are sufficient for strong performance on QA tasks. However, when evaluated on Pile perplexity, we observe significant degradation in perplexity across many sub-domains in Figure 13.7.1. This is likely because synthetic data is very clean containing few special characters and being highly structured. In contrast several sub-domains of the Pile such as OWT, and Hackernews have such special tokens. On domains such as Philpapers and Gutenberg, we observe that dropping real C4 text from the pre-training data, and training on synthetic documents alone drops performance significantly (increase in perplexity). This is once again attributed to the fact that synthetic data does not contain certain 'tags' and 'styles' that are prevalent in real data scrapes, and emphasized how T-MARS is a better strategy than pre-training on synthetic data alone. In terms of performance on zero-shot tasks, the presence of real data helps improve zero-shot performance in Tables **??**. Since these tasks contain well-written Q/A pairs, the effect is not as prominent as on Pile.

(a) Cosine similarity Medium synthetic data

(b) Cosine similarity QA synthetic data

Figure 13.4: Comparison between synthetic and real data from the C4 corpus showing that synthetic data maintains semantic meaning compared with the real C4 data and primarily changes style for (a) medium rephrases of C4, and (b) QA rephrases of C4.

RQ3: Is there data leakage from the rephrase model to the trained model? We investigate whether our synthetic data maintains similar semantic meaning while being stylistically different from the original C4 data and matching the style of different PILE domains. We start by comparing pairs of examples of synthetic and real data to confirm the performance gain is not attributed to knowledge leakage from the rephrase models. We subset the first 1000 samples from each dataset.

We show the cosine similarity of the sentence embeddings from a pre-trained BERT model trained with SimCSE objective [Gao et al., 2021] for medium and qa prompts in Figure 13.4(a) and (b). When computing similarity, we remove outliers. Figures with distributions use a gaussian Kernel Density Estimator (KDE) to construct distributions for statistics from 1000 values. The cosine similarity of real-synthetic pairs is higher than several baselines including two random real samples from C4, a continuation baseline which computes cosine between the first half of a sample and the full sample, and cosine similarity between the first and second half of the same sample. High similarity indicates that rephrases maintain similar meaning to their real counterparts without adding information.

13.6 Rephrase Examples from C4 Corpus

We show one sample document from the C4 corpus below, and then its corresponding rephrase in Medium and Q/A style when using Mistral-7B as the rephraser. More such examples are provided in Appendix H. For the Medium style rephrasing, we ask the LLM to rewrite the sentence in high quality English such as Wikipedia, whereas for the Q/A style rephrase, we ask the LLM to rewrite the document in conversational question-answering format. The detailed prompts are in Appendix H.

Original

```
First round on stress at work survey.  Answering the questionnaire
is voluntary and all answers will be saved anonymously.  Please fill
in this questionnaire only if you have some work experience, part-or
full time.  Otherwise, you will not be able to answer some of the
questions!  Here is a the link to all language version.
```

Medium Style

```
In the initial stage of the survey on work-related stress,
participation is optional, and all responses will be kept
confidential.  It is important to note that only those with
work experience, whether part-time or full-time, will be able
to accurately respond to certain questions.  To complete the
questionnaire, please use the following link, which is available in
multiple languages.
```

Q/A Style

```
Question:  Is participating in the stress at work survey mandatory?
Answer:  No, it is voluntary.  Question:  Will all answers be saved
anonymously?  Answer:  Yes, they will.  Question:  Who is eligible
to take the questionnaire?  Answer:  Only individuals with work
experience, whether part-time or full-time.  Question:  What happens
if someone is not eligible to take the questionnaire?  Answer:  They
will not be able to answer some of the questions.  Question:  Can you
provide a link to all language versions of the questionnaire?  Answer:
Yes, here is the link.
```

13.7 Properties of Synthetic Corpus

To understand the properties of synthetic data generated from the rephrase model that lead to better pre-training performance, we compare the semantic similarity, syntactic complexity, and diversity between synthetic data, C4 data, and data from the PILE. Our primary focus is answering the following questions about synthetic data: (i) Do models trained on synthetic data perform better due to information leakage from the rephrase model? (ii) Does the rephrase model accurately capture multiple styles? (iii) What attributes of synthetic data make it high quality? Our investigation helps address what data is beneficial for better generalization to specific domains, and quantify the importance of data variability and quality.

(a) Cosine similarity Medium synthetic data **(b)** Cosine similarity QA synthetic data

Figure 13.5: Comparison between synthetic and real data from the C4 corpus showing that synthetic data maintains semantic meaning compared with the real C4 data and primarily changes style for (a) medium rephrases of C4, and (b) QA rephrases of C4.

13.7.1 Experimental Setup

We take a subset of the first 1000 samples from each of the datasets. For synthetic comparisons with real C4 data, we take pairs of samples, while for PILE subsets, we take the first 1000 samples from the test subset. When computing dataset quality statistics, we remove outliers. When the number of samples from the PILE subset was fewer than 1000, we split samples. Figures with distributions use a gaussian Kernel Density Estimator (KDE) to construct distributions for statistics from 1000 values.

In this section, we investigate whether our synthetic data maintains similar semantic meaning while being stylistically different from the original C4 data and matching the style of different PILE domains. We start by comparing pairs of examples of synthetic and real data to confirm the performance gain is not attributed to knowledge leakage from the rephrase models. We show the cosine similarity of the sentence embeddings from a pre-trained BERT model trained with SimCSE objective [Gao et al., 2021] for medium and qa prompts in Figure 13.5(a) and (b). The cosine similarity of real-synthetic pairs is higher than several baselines including two random real samples from C4, a continuation baseline which computes cosine between the first half of a sample and the full sample, and cosine similarity between the first half and second half of the same sample. High similarity indicates that the re-phrases maintain similar meaning to their real counterparts without adding information. We also compare the similarity of synthetic rephrases and actual rephrases using the MRPC corpus in Figure 13.5(c). Synthetic rephrases have similar cosine similarity as true rephrases according to the MRPC corpus.

Chapter 14

BeyondWeb

Based on Pratyush Maini et al.. Preprint

Abstract

As web-scale pretraining encounters a data wall, synthetic data emerges as a complementary source for scaling quality. We compare generator-driven and source-rephrasing paradigms and study when and how synthetic data helps. Our analysis highlights key axes—data quality, style, and diversity—and motivates BeyondWeb, a targeted rephrasing strategy that improves information density and coverage at practical cost. We demonstrate consistent gains across model sizes and training budgets, and outline a principled recipe for scalable, effective synthetic data curation.

14.1 Introduction

Until 2024, breakthroughs in language modeling followed a predictable recipe: train ever-larger models on exponentially larger internet-scraped datasets. However, as the scale of data collection ballooned into the trillions of tokens, the field began to encounter a *data wall*, beyond which data of high information-density becomes prohibitively scarce. The returns on collecting more internet data rapidly diminished, pushing researchers toward exploring alternative paradigms.

Synthetic data has emerged as a powerful complement to scarce, high-quality web text. Initial evidence from **Tiny Stories** [Eldan and Li, 2023a] showed that targeted prompting of large language models can generate data suitable to train small language models from scratch. Follow-up work, most notably Microsoft's **Phi** family [Li et al., 2023d] and the open-source **Cosmopedia** corpus [Ben Allal et al., 2024], demonstrated that sub-2B models jointly trained on synthetic and raw data can outperform much larger baselines. We refer to this practice of using large models to generate training data *de novo* as the **generator-driven** approach.

Generator-driven approaches, while powerful, are ultimately limited by the cost and idiosyncrasies of the large generator models they rely on, such as GPT-4 [Maini,

2023]. To address these limitations, the Web Rephrase Augmented Pre-training **(WRAP)** paradigm [Maini et al., 2024d] and its refinement in **Nemotron-CC** [Su et al., 2024] developed what we refer to as the **source rephrasing** approach. In these works, rather than prompting a large LLM to create knowledge de novo, small LLMs are used to rephrase existing web data into higher-quality, task-aligned formats (e.g., Q&A pairs, instructional passages). This approach achieves superior coverage and diversity at lower compute costs, establishing synthetic rephrasing as a practical solution for pretraining-scale data generation.

The success of synthetic data methods has led research organizations to pour substantial compute resources into creating ever larger corpora of rephrased synthetic data [KimiTeam et al., 2025, Hui et al., 2024, xAI, 2025]. Most recently, synthetic data was specifically highlighted as a key innovation in GPT-5's development [OpenAI, 2025]. While we know that synthetic pretraining data *can* work, we still lack a comprehensive scientific understanding of the factors that determine *when* and *how* synthetic pretraining data works. Our work systematically addresses these questions and highlights the challenges of scalably generating high-quality pretraining data by conducting experiments across model scales up to 8B parameters. Specifically, we address the following:

How Does Synthetic Data Provide its Benefits?

- First, we assess whether the model quality improvements imparted by the *generator-driven* paradigm can be explained as distillation of a teacher model's knowledge into higher per-token information density data. We find that simple summarization prompts in the *source-rephrasing* paradigm can match the performance of generator-driven approaches such as Cosmopedia (Sec. 14.4.2).

- Next, we ask whether synthetic data can actually breach the *data wall*. We show that in a data-constrained setting, naive approaches such as simple continuations (of existing web data) provide limited accuracy improvements over just repeating data. However, thoughtfully-created data that *fills in* the distributional gaps of the web data has much larger benefits (Sec. 14.4.3).

What and How to Rephrase?

- First, data **quality** matters: rephrasing high-quality data provides benefits over using lower-quality sources, but high-quality input data alone is not enough to yield the highest quality synthetic data (Sec. 14.4.4).

- Second, distributional **style** matching is important. Web data contains only 2.7% conversational content, despite chat being a major inference use case. Style matching improves performance in downstream tasks, but the benefits saturate quickly with the proportion of conversational data, indicating that it is not a sufficient solution. (Sec. 14.4.5).

- Third, **diversity** in generation strategies is critical to leverage continued benefits from synthetic data when scaling to large training budgets of trillions of tokens (Sec. 14.4.6).

Who Should Rephrase?

- Synthetic data benefits are largely consistent across different rephraser families, and rephraser quality does not predict synthetic data quality, indicating that rephrasing is a generic capability and rephraser model efficacy is difficult to predict.(Sec. 14.4.7).

- We observe diminishing returns when increasing rephraser size beyond 3B parameters, with 8B LLMs achieving only marginal gains over 3B. This validates that (given the right recipe) effective synthetic data generation doesn't necessarily require massive computational resources (Sec. 14.4.8).

Motivated by these insights, we introduce BeyondWeb, a synthetic pretraining data generation paradigm that leverages targeted document rephrasing to yield diverse, relevant, and information-dense synthetic data. On 8B models trained for 180B tokens, we outperform RedPajama [Weber et al., 2024] and the high-quality synthetic subset of Nemotron-CC (Nemotron-Synth) by +7.1pp and +2.6pp, respectively, while achieving training speedups of 7.7x and 2.7x (see Figure **??**). Futhermore, 3B model trained on BeyondWebsurpasses the performance of strong 8B baselines such as Cosmopedia. This establishes a new **pareto frontier** for the accuracy-efficiency trade-off in LLM pretraining. Our results also indicate a promising scaling trend: the benefits of our approach show consistent gains across model sizes, achieving (+3.1pp, +2.0pp, +2.6pp) over Nemotron-Synth at 1B, 3B, 8B parameters respectively.

We note that BeyondWebis one part of our full curation platform, which was used to curate the 7T token pretraining dataset for ArceeAI's AFM4.5B model [Atkins, 2025]—and combining BeyondWebwith our full curation platform obtains even better results. The strength of AFM4.5B demonstrates the efficacy of BeyondWebnot just in the controlled experimental settings presented here, but also as part of an end-to-end curation pipeline for generating foundation-scale pretraining datasets for production models.

Overall, our work demonstrates that generating high quality synthetic data is not a trivial task, and that there is no singular silver bullet solution for obtaining the highest quality synthetic data. Naive approaches to synthetic data generation can provide little benefit or even some detriment, while also incurring substantial computational cost for generation. In contrast, the right synthetic generation approach can yield exceptional model quality improvements, as demonstrated by our BeyondWebresults. Getting synthetic data right requires careful consideration of numerous factors, including data selection

strategies, generation methodologies, diversity preservation, and quality control, making it a challenging endeavor that demands rigorous science and practical expertise.

14.2 A Tale of Two Approaches for Synthetic Pretraining Data

14.2.1 The Generator-Driven Paradigm: Creating Knowledge from Models *de Novo*

The **generator-driven** approach to synthetic data uses large-scale models to generate training data from scratch, encapsulating the knowledge embedded within these models. By generating data from a pretrained model, researchers have been able to substantially improve the performance of the next generation of models. Notably, the seminal work on **Tiny Stories** [Eldan and Li, 2023a] demonstrated that high-quality, carefully prompted synthetic data, such as simplified narratives crafted by GPT-4, could effectively train small language models from scratch. This result defied the prevailing trend of scaling models and web data, sparking a paradigm shift and paving the way for increased use of synthetic data for pretraining small, performant models.

Soon after, the Phi model family [Li et al., 2023d] advanced the generator-driven approach by training small (<2B parameter) models jointly on synthetic data and raw web data, and were able to outperform models up to 10× larger using a fraction of the training compute. **Cosmopedia** [Ben Allal et al., 2024] furthered this direction by introducing a large-scale, open-source synthetic dataset, generated by leveraging open-source LLMs and prompting it with a diverse set of curated seed topics. However, this paradigm, which relies on using a powerful existing model as a knowledge bank to generate synthetic data, is constrained by the high computational cost of accessing state-of-the-art generators like GPT-4, making it inaccessible to many researchers. Furthermore, these approaches do not scale well and are susceptible to model collapse [Shumailov et al., 2024, Briesch et al., 2024, Guo et al., 2024], as they inherit the knowledge and biases of the generator, potentially resulting in limited diversity, coverage gaps, and hallucinated content [Maini, 2023].

14.2.2 The Source Rephrasing Paradigm: Enhancing Existing Knowledge

Motivated by the shortcomings of the generator-driven approach, Maini et al. [2024d] recently proposed an alternative synthetic generation methodology: **Web Rephrase Augmented Pre-training (WRAP)**. WRAP leverages existing web documents and employs

smaller models to rephrase this content into higher-quality, structured and/or targeted formats. Conditioning on existing data significantly reduces dependence on expensive large models while enriching the training corpus with natural, diverse, and topical language. Maini et al. [2024d] were able to speed up pretraining by more than 3× through the use of strategic rephrasing. Of note, Allen-Zhu and Li [2023] concurrently worked on rephrasing articles during pretraining on a toy task of author biographies, and showed such rewriting enhanced model performance. Recently, **REWIRE** [Nguyen et al., 2025] proposed rewriting medium-quality documents into higher-quality ones, which would otherwise be filtered out by classifiers, thereby augmenting the high-quality pool. We refer to this collection of methodologies as the **source rephrasing** paradigm.

The source rephrasing approach has gained considerable traction in industry. Projects such as Nvidia's Nemotron-CC [Su et al., 2024], StabilityAI's multilingual augmentation efforts [Pieler et al., 2024], and Microsoft's Phi-4 [Abdin et al., 2024a] have all built on this paradigm, emphasizing the value of augmenting real-world internet data. This highlights a critical ideological shift toward data-driven rather than model-driven synthetic data generation. This approach effectively combines the advantages of broad, naturally occurring knowledge from the internet with controlled style and format enhancements. In 2025, rephrasing has become the dominant paradigm with state-of-the-art LLMs like Kimi K2 [KimiTeam et al., 2025], Qwen-2.5 [Hui et al., 2024], Grok [xAI, 2025], and GPT-5 [OpenAI, 2025] reporting substantial use of and/or meaningful gains from source rephrasing.

14.2.3 Synthetic Data Beyond Pretraining

The utility of synthetic data extends beyond pretraining and into other phases of model development, including fine-tuning and evaluation. Synthetic data can play a pivotal role in addressing specific task-oriented training objectives, aligning model behavior, reducing toxicity, and improving generalization on targeted downstream tasks. Techniques such as instruction-tuning, Chain-of-Thought prompting, and data backtracing emphasize synthetic data's potential beyond pretraining by directly influencing downstream capabilities, such as reasoning [Lu et al., 2024b], alignment [Li et al., 2024d, Wang et al., 2024c], and reducing model hallucinations [Jones et al., 2024]. However, the relative lack of synthetic data research for pretraining compared to post-training and the potential magnitude of impact from successful pretraining improvements motivated us to focus exclusively on synthetic data for pretraining in this work.

14.3 Introducing BeyondWeb

We introduce BeyondWeb, a synthetic data approach that leverages grounding and diversity to substantially improve language model pretraining efficiency. It combines the broad coverage of web-scale corpora with strategically-generated content that fills critical gaps, particularly in underrepresented styles, formats, and topics. BeyondWebemploys diverse generation strategies, including format transformation (e.g. converting web content into question–answer pairs), style modification (e.g. enhancing pedagogical tone), and content restructuring to improve information density and accessibility. This enables the creation of models better aligned with real-world language use and downstream task demands.

Datasets. We compare BeyondWebagainst state-of-the-art public synthetic pretraining datasets and methodologies, as well as a non-synthetic open web dataset, RedPajama (RPJ) [Weber et al., 2024]. We chose RPJ because it is a well-established baseline that has minimal curation applied to it, allowing us to robustly assess the effects synthetic data. For comparisons with other synthetic datasets, we adopt a mixed curation strategy: 60% of the training tokens are sourced randomly from RPJ, while 40% come from synthetic data. We evaluate against three synthetic baselines:

- **Cosmopedia** [Ben Allal et al., 2024] contains over 39 million textbooks, blog posts, and stories generated by Mixtral-8x7B-Instruct-v0.1, using web-derived prompts covering a predefined set of topics. We use the latest version, Cosmopedia-v2, which contains approximately 27 billion tokens. When more tokens are needed for our experiments, we repeat the dataset to compensate. While repetition is not ideal because it potentially reduces the effective per-token quality of the dataset, we consider this a general limitation of the generator-driven paradigm that stems from the cost and difficulty of prompting and generation.
- **WRAP** [Maini et al., 2024d] rephrases web content into various formats, with question-answering being the most performant style. For our comparison, we use the RedPajama dataset as the source corpus for rephrasing, and the Llama-3.1-8b-Instruct LLM for rephrasing.
- **Nemotron-Synth** [Su et al., 2024] is the high-quality subset of synthetic data in Nemotron-CC. The data was generated by applying diverse rephrasing prompts to high-quality, classifier-selected inputs. We chose the subset of Nemotron-CC synthetic data derived from high-quality input data, which we refer to as Nemotron-Synth, instead of the full Nemotron-CC synthetic data because Nemotron-Synth is an extremely competitive

baseline and sufficiently large for our long-horizon experiments. Nemotron-Synth contains 1.5 trillion tokens, so we randomly sample a subset while maintaining the original proportions across different data subsets.

We apply our BeyondWebapproach to a high-quality subset of DCLM [Li et al., 2024a] selected using the methods described in DatologyAI et al. [2024].

Synthetic Generation Infrastructure. Generating foundation-scale synthetic pretraining data presents a formidable engineering challenge. We addressed this by building a massively scalable data curation pipeline that collects billions of high-quality documents suitable for rephrasing (as detailed in DatologyAI et al. 2024) and then efficiently performs large-scale parallelizable synthetic rephrasing over the resulting trillions of tokens. Furthermore, because our curation platform is not only central to our internal research but also serves as our product, it was designed to be deployable across heterogeneous customer environments. Our initial setup used Slurm on an AWS Hyperpod cluster of H100s, but this system was rigid, slow to iterate on, had deployment constraints, and made it difficult to track experiment choices. This motivated us to move to a solution using Ray and vLLM on Kubernetes. The new setup supports heterogeneous clusters and various GPU types, scales efficiently on Kubernetes, and integrates seamlessly into our existing Ray and Spark curation pipeline. In addition to facilitating deployment and reducing cost and complexity, this also enables us better end-to-end experiment tracking. This infrastructure shift has transformed our synthetic generation system into a flexible, scalable, and production-ready platform that simultaneously accelerates research, streamlines deployment, and powers customer-facing workloads. We look forward to detailing our synthetic data generation infrastructure in a follow-up release.

Training setup. We train three sizes of LLMs: a 1B parameter model trained on 1 trillion tokens, a 3B model trained on 180 billion tokens, and an 8B model trained on 180 billion tokens. We use LLaMA-3.2 architecture [Grattafiori et al., 2024] for the 1B and 3B models, and LLaMA-3.1 architecture for the 8B model. Additional training details can be found in Appendix I. We did not extensively tune hyperparameters other than an early search on the baseline RPJ data, as our goal was to improve model performance with data interventions alone.

Evaluation setup. We evaluated the models on 14 benchmark tasks (enumerated in Appendix I) using both 0-shot and 5-shot prompting and report average accuracy across all settings and tasks. Multiple-choice questions are assessed using a relative scoring method,

as described in Hugging Face's analysis of the OpenLLM leaderboard and referred to as "cloze form" (CF) in Gu et al. [2025]. This approach compares the probabilities assigned by the model, restricted to the set of valid answer choices.

Performance improvements across scales. BeyondWebdemonstrates consistent improvements across all evaluated model sizes, as shown in Figure **??**. At 1B parameters, we achieve 57.4% average accuracy (+6.7pp over RPJ), establishing a strong foundation. The performance gains become more pronounced as scale increases: at 3B parameters, we reach 60.8% accuracy (+7.3pp over RPJ), and at 8B parameters, accuracy rises to 63.7% (+7.1pp over RPJ). BeyondWebalso consistently outperforms Nemotron-Synth, our strongest synthetic pretraining data baseline, at all scales (+3.1pp, +2.0pp, +2.6pp at 1B, 3B, and 8B, respectively). These results demonstrate that our synthetic data approach remains effective across scales and continues to offer substantial benefits in larger models.

Train Models 7.7x faster. BeyondWebenables substantial computational savings during pretraining, as shown in Figure **??**. For the 8B model, BeyondWebmatches or exceeds RedPajama's 180B-token performance in just 23.2B tokens (7.7× speedup) and Nemotron-Synth's 180B-token performance in only 66.2B tokens (2.7× speedup). Faster convergence directly reduces training cost, energy consumption, and iteration time, affording more experiments within a fixed compute budget. This efficiency is valuable not only for large industry labs aiming to optimize costs but also for relatively smaller organizations constrained by infrastructure, thereby helping to democratize access to high-performance LLMs.

Establishing a new pareto frontier for synthetic data. Our results establish a new Pareto frontier for the speed-accuracy trade-off in language model pretraining, as visualized in Figure **??**. Remarkably, our 3B model (60.8% accuracy) outperforms all but the strongest of the 8B baseline models trained for the same number of tokens (180B), demonstrating that high-quality synthetic data can achieve superior results with dramatically fewer parameters. This highlights the scalability and headroom unlocked by high-quality synthetic data, enabling stronger models at lower computational cost and challenging the conventional reliance on ever-larger architectures.

Consistent improvements across tasks. As presented in Table **??**, BeyondWebachieves the highest average accuracy at every model scale and also outperforms baselines on 13, 12, and 13 out of 14 tasks, in 1B, 3B, and 8B models, respectively. This demonstrates that

the performance gains are not confined to a few standout benchmarks but are broadly distributed, reflecting the generalizability enabled by high-quality, diverse synthetic data.

We now examine the factors that contribute to the strength of synthetic data approaches such as BeyondWeband the obstacles to generating high-quality synthetic pretraining data.

14.4 Systematically Evaluating Synthetic Data

Despite the ready availability of web data for pretraining, training models solely on internet-sourced text presents several challenges. First, **data redundancy** is a major issue. Web data is finite, and modern pretraining budgets are beginning to exceed the scale of high-quality web data. Unfortunately, repeating data leads to diminishing returns and overfitting [Muennighoff et al., 2023, Goyal et al., 2024], and thus is not a substitute for new data. The second challenge is **style mismatch**, which describes the stylistic differences between the data models are trained on and the data they encounter at test time. The top three most common styles of web data are personal blogs, product pages, and news articles [Wettig et al., 2025], whereas deployed models predominantly interact in conversational or instructional formats [Maini et al., 2024d]. The third challenge is that **knowledge bottlenecks** emerge in underrepresented domains, making models blind to certain topics. Finally, **training efficiency** can be significantly improved if data is structured to maximize per-token knowledge utility rather than merely increasing token count.

These challenges highlight the need to move beyond web data by generating synthetic data. However, not all synthetically generated data are equally effective. Through extensive experimentation and systematic ablations, we examine how synthetic data can help overcome data limitations, better align training distributions with downstream use-cases, and enhance both knowledge efficiency and diversity. Our systematic evaluation reveals several key findings, structured around three central themes: the capabilities of synthetic data, effective techniques for rephrasing, and the properties of generator models. Table 14.1 summarizes these insights along with section references.

14.4.1 Experiment Setup

As a baseline for the following sections, we use a subset of the RedPajama dataset [Weber et al., 2024]. To construct high-quality splits from RedPajama, we follow the method from DatologyAI et al. [2024] **(HQ Web)**. Unless otherwise specified, all ablations train a Llama-3.2-1B model on a total budget of 20 billion tokens, with a 50:50 split between real (HQ Web) and synthetic data. Unless the experimental ablation demands otherwise, all

Finding	Section
What Does Synthetic Data Really Do?	
Are Generator-Driven Approaches Approximated by Summarization? Simple summarization approaches that increase per-token information density achieve similar performance as Cosmopedia. But carefully-crafted source rephrasing approaches can substantially outperform both.	Sec. 14.4.2
Surpassing the Data Wall: Synthetic data generation must be thoughtful in order to breach the data wall. Naive strategies can not break the data wall.	Sec. 14.4.3
How to Rephrase: Methods & Techniques	
Quality Matters: Synthetically rephrasing high-quality web data offers larger performance gains than using low-quality web data. Nevertheless, high-quality input data alone is insufficient for generating the highest quality synthetic data.	Sec. 14.4.4
Style Matters: Upsampling natural conversational data improves downstream task performance, though these improvements are modest and show diminishing returns with the proportion of conversational data.	Sec. 14.4.5
Diversity Matters: As we scale synthetic data to 1T tokens, the importance of diversity in synthetic data generation is critical to avoid diminishing returns.	Sec. 14.4.6
Who Should Rephrase: Generator Properties	
Model Family Robustness: Synthetic data benefits are largely consistent across generator model families, and generator model quality is not predictive of synthetic data quality. Selecting a good rephrasing model may be straightforward as most work well, but selecting the best rephrasing model is not.	Sec. 14.4.7
Generator Size Saturation: While larger generators help, improvements plateau beyond 3B parameters with diminishing returns beyond this scale.	Sec. 14.4.8

Table 14.1: Summary of key findings organized by theme. Each finding represents a core insight from our systematic evaluation of synthetic data generation strategies.

synthetic data generation approaches rephrase the **same** 10 billion tokens of HQ Web data that are already a part of the final 20 billion data mixture. This means that the total *source knowledge* from the internet is fixed at 10 billion tokens. For a fair baseline that controls for the amount of new knowledge, **RPJ-HQ** baseline also sees the same 10 billion tokens, but twice during its training. For other curated datasets, such as Cosmopedia [Ben Allal et al.,

2024], we randomly sample 10 billion tokens from their corpus. All evaluations report the average performance across 0-shot and 5-shot settings on a benchmark suite of 14 tasks enumerated in Appendix I.

14.4.2 Research Question (RQ) 1: Are Generator-Driven Approaches Approximated by Summarization?

One hypothesis that may explain the benefits of synthetic data is that it effectively increases the **per-token information density** of training data. Rather than merely generating more tokens, the efficacy derives from restructuring and distilling existing knowledge into more compact and informative representations. This hypothesis particularly prevails in the generator-driven paradigm, in which a large LLM is seeded with various topics of interest, and asked to create textbook-style material based on its parametric knowledge. Intuitively, when a model writes a book on a specific topic, say *laws of motion*, it distills all the information it has read about it on the internet into a much more condensed representation. To test this hypothesis, we compare a generator-driven approach (Cosmopedia), with a simple rephrasing approach that merely attempts to increase the information density of the internet by *summarizing* it into more compact, high-quality content.

Summarization Prompt

Summarize the following text. Directly start with the summary. Do not say anything else.

Experiment Design. We consider two approaches: (1) **Cosmopedia**, which prompts a large model to explain or teach concepts, reorganizing knowledge into compact, pedagogically optimized outputs. Cosmopedia uses an 8x7B model to generate high-quality data conditioned on seed topics. (2) **Summarization**, which compresses existing text using a prompt that aims to preserve essential information while reducing length. This approach leverages an 8B-parameter model with a simple prompt to summarize source content. Both approaches aim to condense learning-relevant content into fewer tokens, enabling more efficient training under fixed budgets. Following the setup in Section 14.4.3, we allocate 10 billion tokens for synthetic data and 10 billion tokens for the original dataset during training.

Results and Observations. Our analysis in Figure **??** reveals that even simple distillation strategies can achieve substantial performance improvements.

1. **Simple summaries match generator-driven methods**: The summarization approach

Figure 14.1: Knowledge transfer effectiveness across different synthetic data approaches. The yellow line represents **Cosmopedia (46.8%)**, which uses a generator-driven, sophisticated educational content generation technique and 8x7B model; the cyan line denotes the **Summary (46.8%)** approach, which uses an 8B model and a simple summarization prompt; the gray line denotes the **RPJ-HQ** (no synthetic data) baseline. These results demonstrate that even naive summarization achieves substantial improvements similar to those of Cosmopedia, suggesting distillation works through increased information density rather than complex knowledge transfer.

(46.7%) nearly matches the performance of the more sophisticated and compute-intensive Cosmopedia approach (47.1%). This suggests that much of the benefit of synthetic data arises from basic information condensation. Notably, summarization, which uses a single 8B-parameter model along with information on the internet, nearly matches the performance of Cosmopedia, which relies on a much larger setup of 8×7B-parameter model and manually curated seed topics to guide generation. Importantly, this shows that in the case of Cosmopedia, much of the benefit of the generator-driven paradigm can be achieved using the source-rephrasing paradigm and a simple summarization prompt.

2. **Increasing token information density improves model performance**: The fact that naive summary prompts achieve a +1.2pp improvement over the baseline (45.5%) indicates that increasing per-token information density is one mechanism (among potentially many) by which synthetic data improves pre-training.

3. **Synthetic data is not just knowledge distillation**: While summarization shows similar benefits as Cosmopedia, BeyondWeboutperforms summarization-based approaches with 50.4% accuracy (+3.7pp). This highlights that substantial gains are possible beyond basic distillation and underscores the importance of careful synthetic data generation.

Takeaway: Synthetic Data Is Not Just Knowledge Distillation

We find that a simple summarization-based rephrasing approach is able to nearly match the performance of the much more expensive generator-driven approach of Cosmopedia. This highlights that increasing per-token information density is one mechanism (among potentially many) by which synthetic data improves pre-training. However, BeyondWeboutperforms summarization-based approaches by a clear margin, showing that substantial gains are possible beyond basic distillation, and underscoring the importance of intentional synthetic data approaches.

14.4.3 RQ2: Can Synthetic Data Break the Data Wall?

Can synthetic data effectively compensate for limited high-quality real-world data? This section explores whether there exists a fundamental limit to model performance that cannot be overcome through synthetic data. Our results reveal a surprising finding: **the data wall is surpassable** ***depending on the type of synthetic data***.

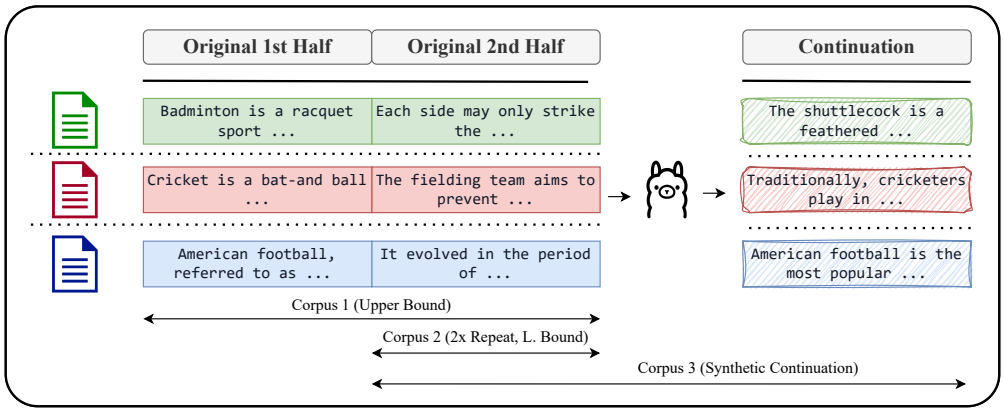


Figure 14.2: Continuation vs. repetition vs. full data upper bound. Illustration of data splitting and corpus construction strategies to enable a controlled setup. The figure shows how our 20 billion token dataset is divided and utilized across different experimental conditions. The top row displays three data segments: **Original 1st Half** (10B tokens of natural web content), **Original 2nd Half** (10B tokens of natural web content), and **Continuation** (10B tokens of synthetic content generated by extending documents from the first half). The example text snippets demonstrate how continuation generates stylistically consistent but novel content. The arrows below indicate corpus composition: **Corpus 1 (Upper Bound)** uses both original halves for full natural data coverage; **Corpus 2 (2x Repeat)** uses only the first half repeated twice; and **Corpus 3 (Synthetic Extension)** combines the second half with synthetic continuations. This experimental design isolates the effects of repetition versus synthetic augmentation when facing data constraints.

Continuation Prompt

Continue the following text in the same style as the original.

Experiment Design. We designed three controlled datasets (see Figure 14.2) to systematically compare different training approaches operating in a data-constrained setting of 20 billion tokens.

1. **Full Data (Upper Bound).** This represents our performance ceiling, utilizing the complete available dataset of 20 billion tokens without any repetition or synthetic augmentation. This corpus serves as the *oracle* baseline, representing the maximum

achievable performance given our data constraint. The figure shows this as a complete, unbroken dataset.

2. **2x Repeat (Lower Bound).** This corpus simulates data scarcity by artificially constraining our dataset to only 10 billion unique tokens, then repeating this exact content twice to reach our full 20 billion token training budget. We split each paragraph at the sentence boundary closest to its midpoint, and only retain the second half of the paragraph. This approach represents a naive solution of repeating data to meet training token requirements, and serves as our lower bound.

3. **Continuation (Naive Synthetic).** Building on the same 10 billion token dataset as the previous strategy, this approach generates the remaining 10 billion tokens through simple model-driven continuation. An LLM (Llama-3.1-8B in this case) receives partial documents from the *second half* of the original dataset and is prompted to generate natural continuations. This represents the most straightforward approach to synthetic data generation, where models extend the available data. The synthetic content attempts to maintain stylistic consistency with the source material. We specifically chose the *second* half of each paragraph in the original data to avoid the confounding effect of the generator model already having seen the full sample during its own pretraining. If the generator model had seen the sample, then it may be able to 'cheat' by generating the true second half when prompted with the first half, and hence approximate the full data upperbound. This would allow us to test whether naive continuation of part of the data can approximate the whole.

Results and Observations. Figure 14.3 offers a visual comparison of average accuracy across models pretrained on different datasets using various augmentation strategies.

1. **Repetition leads to performance degradation**: The 2x Repeat strategy (gray line) achieves only 45.5% accuracy, representing a 0.7pp drop from the full data baseline (46.2%). This demonstrates that naive data duplication leads to diminishing returns as seen in prior works [Muennighoff et al., 2023, Goyal et al., 2024].

2. **Naive synthetic generation provides only modest improvements**: The Continuation strategy (46.2%) offers a small +0.7pp gain over 2x Repeat (45.5%). This matches the Full Data upper bound, suggesting that simple model-generated extensions can potentially compensate for limited quantities of web data. We note that an important confounder in this experiment is that the continuation model has seen a lot more than the 20B

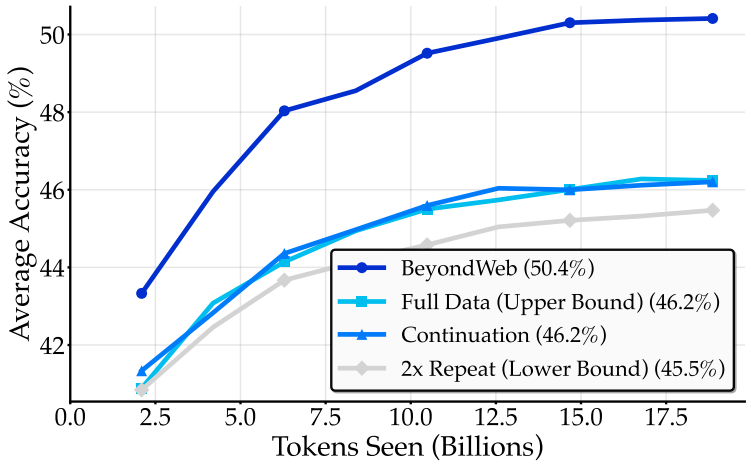


Figure 14.3: Breaking the data wall with strategic synthetic data. Performance comparisons across different data augmentation strategies during training. The dark blue line represents **BeyondWeb(50.4%)** which significantly surpasses all other approaches. The light blue line shows **Continuation (46.2%)**, the cyan line depicts **Full Data Upper Bound (46.2%)**, and the gray line represents **2x Repeat Lower Bound (45.5%)**. The striking visual separation emphasizes BeyondWeb's +4.2pp improvement over the Full Data upper bound. This reflects how intentionality is critical to breaking the data wall with synthetic data, and not just *any* synthetic data will yield benefits.

tokens available for full data experiment. This means that it may be using its parametric knowledge to add 'new knowledge'.

3. **Strategic synthetic data breaches the data wall**: BeyondWebachieves 50.4% accuracy, surpassing the Full Data upper bound (46.2%) by +4.2pp. This finding demonstrates that carefully crafted synthetic data can exceed the performance ceiling of natural data. The data wall is not unsurpassable; it can be broken through strategic synthetic data generation.

Takeaway: Synthetic Data Generation Must be Thoughtful to Improve over Real Data

Simple continuation provides limited improvement over repetition, and may not be able to breach the data wall. Synthetic data must be well designed, as shown by BeyondWeb, which can significantly exceed the performance of training solely on natural data.

14.4.4 RQ3: How Important is the Quality of the Seed Data for Rephrasing?

An important design decision in synthetic data generation involves the quality of both the source material and the generated synthetic content. This raises a fundamental question: *if*

one has limited high-quality data, is it better to use high-quality (but repeated) input data to seed synthetic rephrasing, or leverage lower-quality sources and turn them into high quality?

Experiment Design. We operate under a constraint of 10B tokens of high-quality data, along with an abundant supply of low-quality data. Following the method from DatologyAI et al. [2024], we sample a small subset of high-quality (HQ Web) data from RedPajama. We also use a random sample of the RPJ dataset as our low-quality subset (LQ Web). We then rephrase the LQ Web data to yield LQ Synth and the HQ Web data to yield HQ Synth, and compare training scenarios that use different quality combinations of synthetic and/or web data. We note that we rephrase the *same* HQ web data that we also use in the data mix.

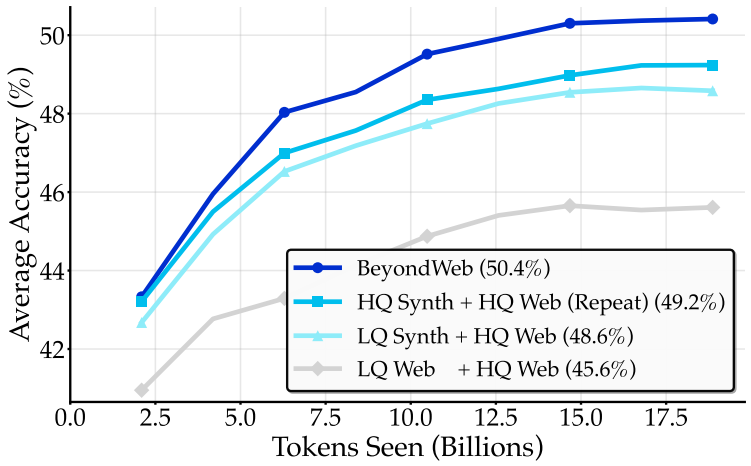


Figure 14.4: Effect of seed quality on rephrasing outcomes. Performance comparison across different quality combinations in training data. **HQ** refers to high-quality web data, LQ refers to low-quality web data. The dark blue line shows BeyondWeb(50.4%), dark cyan shows **HQ Synth + HQ Web (49.2%)**, where the synthetic data are rephrased versions of the HQ web samples, and the light cyan line shows **LQ Synth + HQ Web (48.6%)**. The gray baseline corresponds to **LQ Web + HQ Web (45.6%)**. These results indicate that improving the quality of input data for rephrasing improves the rephrased data, even when there is overlap with the original input data. But improved input data quality alone is inadequate for producing the highest quality synthetic data.

Results and Observations. Figure 14.4 shows two important insights about input data for rephrasing:

1. **High-Quality Seed Data Dominates**: The high performance of the combination of HQ Synth + HQ Web (49.2%) demonstrates that starting with high-quality source material for synthetic rephrasing is crucial. Notably, the BeyondWebperformance at 50.4% shows that there remain other avenues for performance improvement beyond high-quality source data.

2. **Quality Trumps Novelty**: The LQ Synth + HQ Web combination (48.6%) underperforms the HQ Synth + HQ Web data (49.2%), indicating that the quality of seed data for synthetic generation matters more than ensuring complete novelty in the knowledge base. However, it still provides a +3.0pp improvement over the baseline, consistent with the findings of Nguyen et al. [2025].

Takeaway: Quality Synthesis Can Enable Knowledge Amplification

Given limited high-quality data, it is more beneficial to use it as seed data for synthetic rephrasing rather than low-quality data, despite the possible drawback of repeating knowledge from the high-quality sources. Nevertheless, high-quality input data alone is insufficient for generating the highest quality synthetic data.

14.4.5 RQ4: Is Distributional Style Matching Important?

While vast amounts of text data are available online, the natural distribution of web content is different from the types of text typically encountered at inference. For instance, the internet is dominated by personal blogs, news articles, and product pages [Wettig et al., 2025]. However, evaluation use cases typically involve chat, or in-context learning abilities. These can together be stylistically seen as a 'ping-pong' rally between questions and answers, or users and assistants. We investigate the impact of this implicit distribution shift between train and test, and if such stylistic alignment can be one mechanism by which synthetic rephrasing can improve model performance.

Estimating the amount of conversational web data We first estimate the fraction of conversational data in the general web corpus. To do so, we sampled 10k random examples from the RedPajama dataset, and queried the `gpt4o` model to label them as conversational or not. Here conversational refers to various forms of texts which have a 'ping-pong' effect (see Appendix I for prompt). Our analysis indicated that conversational dialogue comprises less than 2.7% of internet text. Yet, the dominant use case for modern language models lies in chat-based applications, such as virtual assistants, customer service bots, and interactive tools.

This discrepancy reveals a gap between the training distribution and real-world model usage. The formats and interaction styles most relevant to deployment, such as multi-turn dialogue, question answering, and instructional content, are significantly underrepresented in naturally occurring data. Synthetic rephrasing provides a mechanism to bridge this gap by generating examples in the specific forms and contexts that matter for downstream applications.

Estimating the impact of conversational web data To systematically evaluate the impact of style alignment, we conduct controlled experiments that vary the proportion of conversational content in our training data while maintaining constant token budgets and data quality. We leverage the style classification filters from Organize the Web [Wettig et al., 2025] to identify naturally occurring conversational content within the RedPajama dataset. From their 20 identified web content styles, we focus on four that exhibit conversational characteristics: `Audio Transcript`, `Customer Support`, `FAQ`, and `Q&A Forum`. Manual inspection confirmed these categories contain the back-and-forth dialogue patterns characteristic of conversational interaction.

As a validation of our previous investigation, this naturally occurring conversational content comprises 3.67% of the RedPajama dataset, roughly aligning with our GPT-based estimate of 2.7% mentioned earlier. We construct three training datasets with varying conversational ratios: 10%, 20%, and 50% conversational content, with the remainder consisting of randomly sampled RedPajama data. Each dataset maintains our standard 20 billion token training budget, enabling direct performance comparisons across different style distributions.

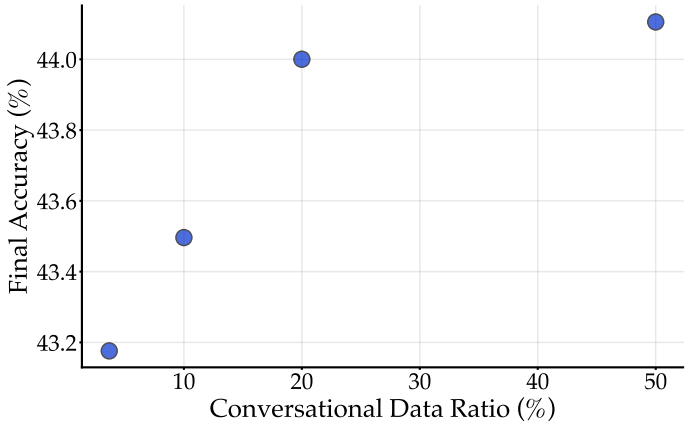


Figure 14.5: Effect of conversational data ratio on final accuracy. The first point corresponds to the **RPJ baseline** (randomly sampled) which contains **3.67% conversational data**. For each of the other points, we **upsampled conversational data** in RPJ to the desired ratio and replaced the remaining portion with randomly sampled RPJ data to achieve the 20B token training budget. Increasing the percentage of conversational data beyond the 3.67% baseline can improve performance by up to 0.9pp at 50% conversational data, but the gains saturate beyond 20%, indicating that style-matching is important, but not sufficient for maximizing synthetic data quality.

Results and Observations. Since our focus is on multi-turn, back-and-forth conversations, for this evaluation we measure only the 5-shot performance of the trained models. We find that increasing the proportion of conversational content in pretraining improves

model performance (Figure 14.5). The first point represents the **RPJ baseline**, which contains 3.67% conversational data from random sampling. For higher ratios, we upsample conversational data within RPJ to the target percentage and replace the remaining portion with randomly sampled RPJ data. Performance increases from 43.2% at the baseline to 44.1% at 50% conversational content, with 10% and 20% conversational data yielding 43.5% and 44.0%, respectively. This trend confirms that increased exposure to conversational patterns during pretraining modestly enhances downstream task capabilities, though gains exhibit diminishing returns.

Takeaway: Distributional Style Matching is Useful but not Sufficient

Conversational content is a small fraction of web data (3.67%), yet chat and in-context learning are the primary inference use cases of LLMs. Upsampling natural conversational data improves downstream task performance, though these improvements are modest and show diminishing returns with the proportion of conversational data. These results indicate that style-matching can improve synthetic data quality, but is not sufficient for maximizing it.

14.4.6 RQ5: How Important is Diversity at Scale?

In the previous section, we found that aligning styles between train and test time has positive but diminishing gains. This finding highlights a critical limitation: while style alignment helps initially, models eventually saturate on uniform synthetic data patterns. This leads us to examine the importance of diversity in synthetic data generation. We investigate a fundamental question: How does synthetic data diversity affect model performance when scaling to trillions of tokens? Specifically, do diverse generation strategies maintain benefits throughout extended training, or do all approaches eventually plateau? Our experiments reveal that different synthetic data generation approaches yield dramatically different scaling outcomes, with approaches that emphasize diversity, exemplified by BeyondWeb, significantly outperforming fixed synthetic data generation techniques.

Experiment Design. To understand the scaling effects of different synthetic data strategies, we analyze training dynamics across model scales through extended training runs. Among the strategies we investigate, Cosmopedia generates a textbook related to a web sample under the generator-driven paradigm, QA Wrap rephrases existing web documents into conversational question-answers. Nemotron-Synth builds on top of WRAP by diversifying the prompts beyond just conversational question-answering, into various styles like

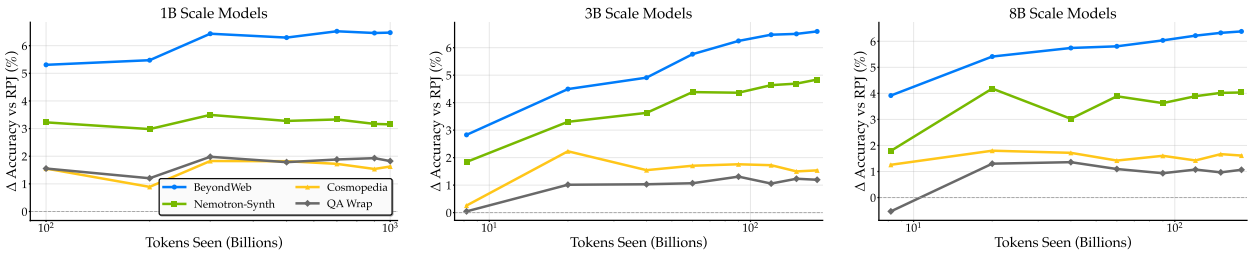


Figure 14.6: Training dynamics across model scales showing performance improvements relative to RedPajama baseline. **Left:** 1B Scale, **Center:** 3B Scale, **Right:** 8B Scale. BeyondWeb(blue) maintains consistent positive slopes across all scales, while QA approaches (cyan) show early gains followed by plateauing effects, particularly evident in larger models. Notably, at 1B scale, models are trained to saturation at approximately 50× beyond Chinchilla optimal compute. Where other baselines begin to overfit and degrade, BeyondWebcontinues to grow in its improvements over RedPajama performance. The sustainable improvement curves demonstrate that diverse generation strategies provide more robust long-term benefits than single-strategy approaches across training regimes. Note that in order to smooth intermediate fluctuations, each data point on the plot is the average accuracy of multiple previous intermediate checkpoints (e.g. the 100B token checkpoint is the average of all checkpoints until 100B, the 200B checkpoint is the average of all checkpoints between 100B and 200B, etc).

MCQ, Yes/No questions, open-ended questions, reading comprehension tasks, logical problem solving tasks, and so on. BeyondWebfurther prioritizes diversity in synthetic data generation, and prioritizes learning patterns that aid learning and information retention.

Results and Observations. The training dynamics reveal fundamental differences between generation strategies as we scale to hundreds of billions of tokens of data, particularly evident across different training regimes. Diversity in synthetic data generation manifests in two critical ways:

1. **Diversity provides an initial performance bump:** Multi-strategy approaches like BeyondWebprovide substantial early gains as models quickly benefit from immediate exposure to diverse formats and styles. We hypothesize this aids learning by making patterns that align with model's final usage, core to its early representations.

2. **Diversity leads to sustained improvements:** Diverse generation strategies continue providing learning benefits throughout extended training, even in overtrained regimes. At the 8B scale, the marginal improvements for BeyondWebover RPJ (y-axis) continue to increase, whereas the benefits for strategies such as Cosmopedia (which generates a fixed style of textbook content) tend to saturate (Figure 14.6). Remarkably, at 1B scale where models are trained approximately 50× beyond Chinchilla optimal compute, BeyondWebmaintains positive performance trends while other baselines begin to overfit and show degraded performance, demonstrating the robustness of diverse synthetic

data against overfitting in extreme training regimes.

Takeaway: Diversity Enables Sustained Learning

While standard synthetic data generation methods provide initial benefits as models adapt to consistent formats, their lack of stylistic diversity leads to diminishing returns. Multi-faceted approaches continue providing valuable learning signals throughout training, whereas single-strategy methods saturate.

14.4.7 RQ6: How Important is the Synthetic Rephraser Model Family?

A key difference between the source rephrasing and the generator-driven approach is that the synthetic data generator is not required to be the 'source' of the knowledge. Instead it only needs to transform the data to make it more useful for pretraining. While existing work like Cosmopedia [Ben Allal et al., 2024] and Phi-1.5 [Li et al., 2023d] requires large generator models, Mixtral-8x7B-Instruct-v0.1 [Jiang et al., 2024] and GPT-4, respectively, we question whether the specific generator family used for rephrasing significantly impacts the quality of the rephrased data.

Experiment Design. We test the importance of generator family by using 4 different generator models: (i) OLMo-2-7B[1] [OLMo et al., 2024], (ii) Phi-4-14B[2] [Abdin et al., 2024b], (iii) Mistral-7B-v0.3[3] [Jiang et al., 2023b], and (iv) Llama-3.1-8B[4] [Grattafiori et al., 2024]. We use the same rephrasing prompt for each model and evaluate the performance of a model trained on the synthetic data generated by each of these generator models. We generated 10B tokens of synthetic data using each of them, combined it with 10B tokens of web data, and trained a model on each of the resulting 20B token datasets.

Results and Observations. Synthetic data rephrased using different model families consistently produces high-quality training data, with the resulting datasets achieving gains ranging from +3.4pp to +4.4pp over the RPJ baseline (45.5%). All model families were within ±1pp of each other, with OLMo-2-7B achieving the highest performance (49.9

We compare the general benchmark performance of each rephraser model (OLMo-2-7B: 59.6%, Llama-3.1-8B: 61.2%, Mistral-7B-v0.3: 66.0%, and Phi-4: 66.6

[1] https://huggingface.co/allenai/OLMo-2-1124-7B-Instruct
[2] https://huggingface.co/microsoft/phi-4
[3] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3
[4] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

These findings suggest that the primary driver of performance is not the generator's general language modeling capabilities, but rather the limited and simpler capability to perform effective rephrasing and data transformation, which appears to be consistent across different model families. This generator-agnostic behavior means that organizations can deploy synthetic data pipelines using available model families without worrying about generator-specific optimizations, thereby reducing overall resource usage and enabling the viability of open-source and permissible models.

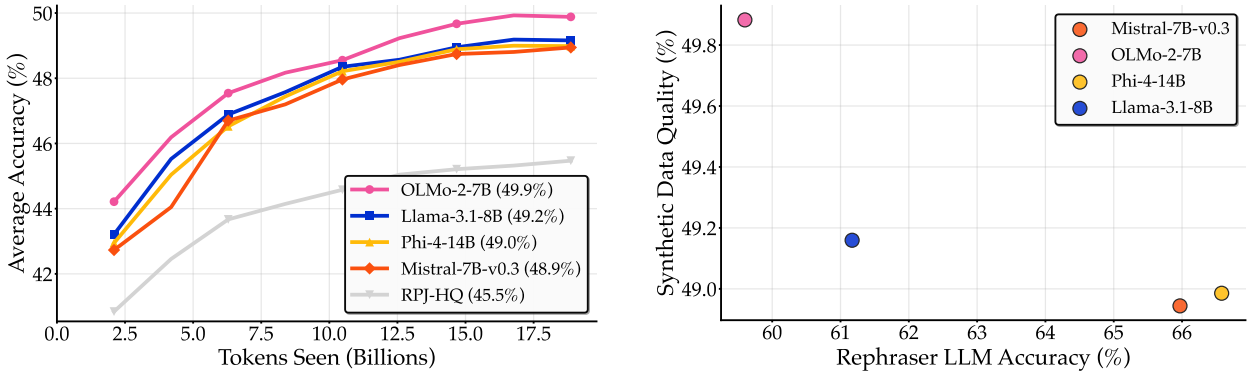


Figure 14.7: Synthetic data benefits are largely consistent across generator model families, and generator model quality does not predict synthetic data quality. **Left:** This plot shows the quality of models trained on synthetic data generated by different models: the pink line represents data generated by OLMo-2-7B (49.9%), the yellow line shows data generated by Phi-4-14B (49.5%), the orange line indicates data generated by Mistral-7B-v0.3 (48.9%), and the blue line depicts data generated by Llama-3.1-8B (49.2%). Models trained on each of these synthetic datasets achieve substantial improvements over the gray baseline representing RPJ (45.5%), with three out of the four models generating synthetic data of nearly identical quality, and OLMo-2-7B generating better data. **Right:** We plot the benchmark performance (on the x-axis) of the rephraser models used for synthetic data generation and contrast with the performance of the LLM trained on their generated data (marked as synthetic data quality on the y-axis). We observe that the benchmark performance of a language model does not positively correlate with its rephrasing capability.

Takeaway: Generator Knowledge is Less Important

Synthetic data benefits are largely robust across generator families, with consistent improvements across Phi-4-14B, Mistral-7B-v0.3, and Llama-3.1-8B, and further improved synthetic data quality from OLMo-2-7B. Additionally, generator model quality is not predictive of synthetic data quality. These results indicate that selecting a *good* rephrasing model may be straightforward as most work well, but selecting the *best* rephrasing model is not.

14.4.8 RQ7: Does Rephraser Size Matter?

Having observed near-invariance in the performance of synthetic data generated by different model families, we now turn to the question of whether the size of the rephraser model affects synthetic data quality. Rephrasing is a much more constrained task than open-ended generation, so it's possible that even small models may be able to generate high-quality synthetic data via rephrasing. This is once again in contrast to the generator-driven paradigm where the generator model is required to be the *source* of the knowledge, and the quality of the synthetic data is heavily dependent on the size of the generator model [Ben Allal et al., 2024].

Experiment Design. To investigate this, we used Llama-3 models of varying sizes, 1B, 3B, and 8B, to generate 10B tokens of synthetic data. We combined the 10B synthetic tokens with 10B tokens of web data, and trained a model on each of the resulting 20B token datasets.

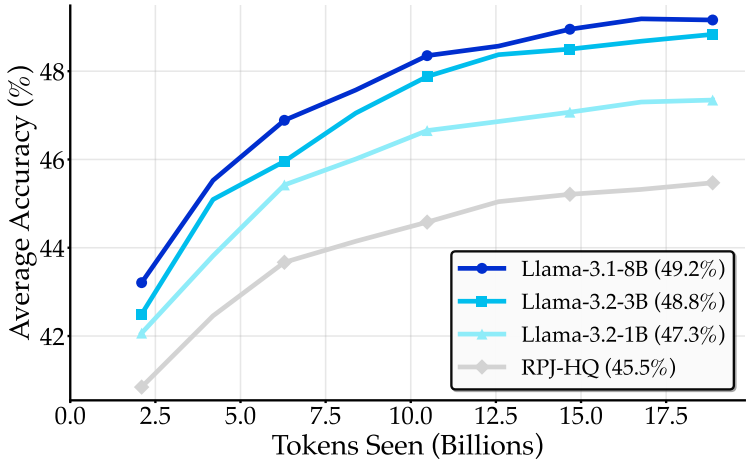


Figure 14.8: Effect of generator model size on synthetic data quality. The plot demonstrates the impact of generator size: the light cyan line shows Llama-3.2-1B generator performance (47.3%), the cyan line represents Llama-3.2-3B generator results (48.8%), and the dark blue line indicates Llama-3.1-8B generator performance (49.2%). The improvement from 1B to 3B (+1.5pp) is substantial, while the gain from 3B to 8B (+0.4pp) shows diminishing returns. The gray baseline at 45.5% represents RPJ performance, emphasizing the consistent improvements achieved by all generator sizes.

Results and Observations. As depicted in Figure 14.8, larger generators consistently produce better synthetic data, yielding substantial downstream performance gains (+3.7pp for 8B, +3.3pp for 3B) over RPJ baseline (45.5

While the 8B model (49.2Larger generators likely offer advantages in knowledge depth, reasoning, and content diversity, which smaller models cannot replicate. However, this

3B capability threshold makes high-quality synthetic data accessible to researchers with limited computational resources. This observation, however, is specific to our setup and may vary under more sophisticated generation methods or on more demanding benchmarks.

Takeaway: Small Models Can Produce High Quality Synthetic Data

The quality of synthetic data increases when increasing generator size from 1B to 3B, then starts to saturate at 8B. The simplicity of rephrasing makes generator size less critical, enabling highly scalable synthetic pretraining data generation even with small models.

14.5 Future Directions

Our work with synthetic data opens several research directions that warrant further investigation:

Scaling Laws and Inherent Repetition. A crucial area for future research lies in understanding the scaling laws specific to synthetic data. Unlike real data, where repetition is explicit and measurable, synthetic data presents a more nuanced form of inherent repetition stemming from the generating model's architecture and the prompting methods used. This intrinsic repetition, governed by the model's parameters and rephrasing strategies, requires new theoretical frameworks and metrics for quantification. The development of such metrics would enable us to better assess the true value and diversity of synthetic data. This understanding is particularly important as we scale up synthetic data generation, as it could reveal fundamental limits or opportunities that differ from those observed in traditional scaling laws for real data.

Democratizing Synthetic Data Generation. An important observation from our work is that the task of rephrasing, at its core, doesn't necessarily require extensive world knowledge. This suggests the possibility of using much smaller models for synthetic data generation, which could significantly reduce computational costs. Future work should explore the minimal model size required for effective rephrasing while maintaining quality, potentially making synthetic data generation more accessible to researchers with limited computational resources.

Alignment with Human Values. Since the synthetic data generation process offers greater control, it creates promising opportunities for developing pretraining method-

ologies that explicitly promote alignment with human values [Korbak et al., 2023]. This aspect of synthetic data training could provide a powerful tool for addressing alignment challenges in language model development, offering an alternative to post-hoc alignment techniques [Maini et al., 2025].

Applicability Across Data Domains. While the present work, like most documented applications of source rephrasing, uses web data as input, the approach is not inherently limited to web data or even the text modality. It can be applied to domain-specific or proprietary data without requiring the rephrasing model to possess domain expertise, making it broadly suitable across diverse settings. Future work should explore applying such techniques to new domains and modalities, using synthetic data generation to help overcome the inherent data wall.

These research directions highlight the potential of synthetic data to not only improve model performance but also to address fundamental challenges in machine learning, from accessibility and scalability to alignment and safety. Future work in these areas could significantly impact how we approach language model training and deployment.

14.6 Conclusion

Pretraining now routinely encounters a data wall: the supply of high-quality, information-dense web text can not keep up with modern training budgets. Synthetic data has emerged as a natural and effective response, yet the scientific understanding of when and how it helps has remained limited. We address this by studying synthetic data for pretraining at scale and introduce BeyondWeb, a rephrasing-driven approach grounded in real web documents and diversified across styles and formats.

Our work identifies three general principles for effective synthetic data generation: prioritize quality over novelty by rephrasing high-quality sources rather than adding low-quality content; align the training data style with the deployment use-case; and maintain diversity in generation to sustain improvement over long training horizons. We show that these benefits are robust across generator families and saturate beyond moderate rephraser sizes, indicating that effective synthetic data generation does not require a particular model architecture or very large generators. However, we also show that the contribution of each of these factors in isolation is often modest, and no single factor is sufficient to yield high-quality synthetic data.

Taken together, these results demonstrate that high quality synthetic pretraining data is challenging to generate, even more challenging to ensure the efficacy of for larger models

and training budgets, and risks being extremely costly to generate. Furthermore, there is no silver bullet for synthetic data, strong outcomes require jointly optimizing many variables. However, all of these challenges and risks are surmountable with thoughtful, scientifically rigorous source rephrasing, as evidenced by BeyondWebsubstantially outperforming other public synthetic pretraining data offerings.

14.7 Contributions and Acknowledgements

Core Research Pratyush Maini and Vineeth Dorna.

steering the ship, wrangling the data, and keeping the Celsius industry thriving.

Core Infra Parth Doshi, Aldo Carranza, Fan Pan, Jack Urbanek and Paul Burstein.

the all-star cast who turned Mazra into code, numbers, and wins.

Leadership Bogdan Gaza, Ari Morcos, and Matthew Leavitt.

the wise captains that made sure we didn't steer into a bad baseline.

Contributors Alex Fang, Alvin Deng, Amro Abbas, Brett Larsen, Cody Blakeney, Charvi Bannur, Christina Baek, Darren Teh, David Schwab, Haakon Mongstad, Haoli Yin, Josh Wills, Kaleigh Mentzer, Luke Merrick, Ricardo Monti, Rishabh Adiga, Siddharth Joshi, Spandan Das, and Zhengping Wang.

the ping-pong maestros who kept latency low and rallies long.

Acknowledgements Gem De Leon and Kristin Reinke for fueling us with Beyond Burgers and unshakable good vibes. Jacqueline Liu and Tiffanie Pham for assembling the all-star cast that made this work possible. Liz Gatapia for the beautiful logo design. Kylie Clement, Jeremy Custenborder and Elise Clark for their sharp-eyed feedback on the draft.

the backstage crew who kept us fed, staffed, and focused.

Chapter 15

Understanding Hallucinations in Diffusion Models

Based on Pratyush Maini et al.. Preprint

Abstract

Colloquially, image generation models based upon diffusion processes are frequently said to exhibit hallucinations—samples that could never occur in the training data. We study a particular failure mode in diffusion models, which we term mode interpolation: diffusion models smoothly interpolate between nearby data modes to generate samples completely outside the support of the training distribution, leading to artifacts that never existed in real data. Through experiments on 1D and 2D Gaussians, we show how a discontinuous loss landscape in the decoder leads to regions where any smooth approximation causes such hallucinations. On synthetic shape datasets, we observe combinations of shapes that never existed. We extend validation to real-world data by explaining images with additional or missing fingers similar to outputs of popular text-to-image models. Finally, we show diffusion models know when they hallucinate: the variance of the predicted trajectory spikes in the final steps. A simple variance metric removes over 95

15.0.1 Hallucination in Diffusion Models

Before formalizing our notions and definitions in Sec. 15.1, let us first consolidate the observation that has been loosely labeled as 'hallucination' until now. To illustrate this phenomenon, we design a synthetic dataset called Squares-Triangles, and train a diffusion model to learn its distribution.

Squares-TrianglesSetup Consider a dataset consisting of black and white images that contain three shapes: triangle, square, and pentagon. Each image in the dataset is 64x64 pixels in size and divided into three (implied) columns. The first, second, and third columns contain a triangle, square, and pentagon, respectively. Each column has a 0.5 probability of

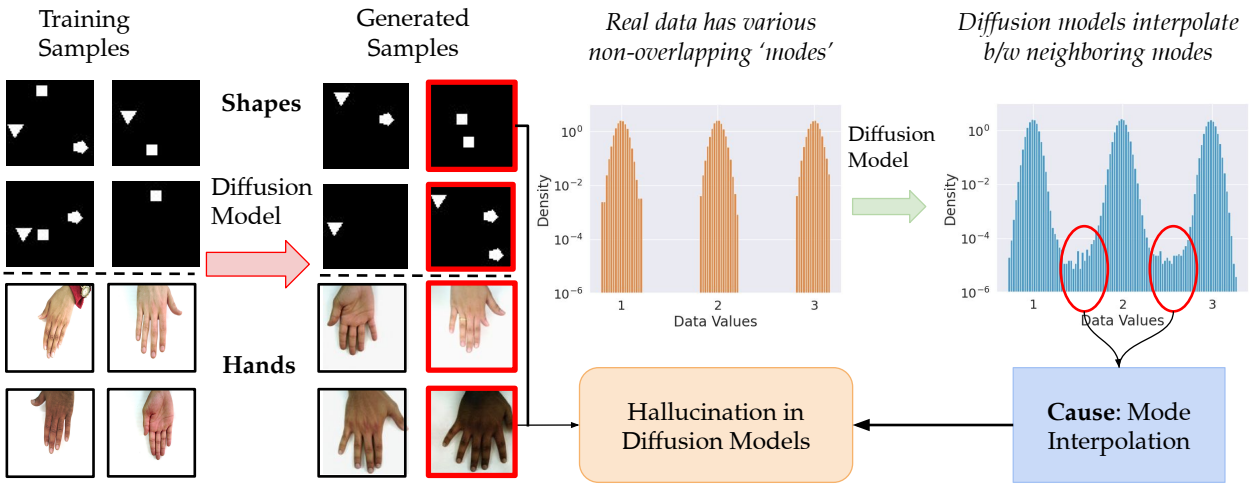


Figure 15.1: Hallucinations in Diffusion Models: Original Dataset (Left) & Generated Dataset (Right). (Top) The original dataset consists of 64x64 images divided into three columns, each containing a triangle, square, or pentagon with a 0.5 probability of the shape being present. Each shape appears at most once per image. The generated dataset created using an unconditional DDPM includes some samples (*hallucinations*) with multiple occurrences of the same shape that is unseen in the original dataset. (Bottom) We also train a ADM [Nichol and Dhariwal, 2021] on a dataset of high-quality images of human hands and show that the diffusion model generates hallucinated images of hands with additional fingers.

containing the corresponding shape. A representation of this setup is shown in Fig 15.1. It is important to note that in this data generation pipeline, each shape is present at most once in each image.

Observation We train an unconditional Denoising Diffusion Probabilistic Model (DDPM) [Ho et al., 2020] on this toy dataset with $T = 1000$ timesteps. We observe that the DDPM generates a small fraction of images that are never observed in the training dataset, nor a part of the 'support' of the data generation pipeline. Specifically, the model generates some images that contain two occurrences of the same shape, as shown in Fig 15.1. Furthermore, when the model is iteratively trained on its own sampled data, the fraction of these occurrences increases significantly as the generation process progresses.

Inspired by these observations and their implications, we will perform experiments through the rest of this work to formalize what we mean by hallucinations (Sec. 15.1), why do they occur (Sec. 15.2), how can we mitigate them (Sec. 15.3), and what are their implications for real-world datasets (Sec. 15.4).

15.1 Definitions and Preliminaries

Let $q(x)$ be the real data distribution. We define a forward process where Gaussian noise is iteratively added at each timestep for a total of T timesteps. Let $x_0 \sim q(x)$, and x_t be the perturbed (noisy) sample after adding t timesteps of noise. The noise schedule is defined by $\beta_t \in (0, 1)$, which represents the variance of Gaussian (added noise) at time t. For large enough T, $x_T \sim \mathcal{N}(0, \mathbf{I})$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}); \qquad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{15.1}$$

In the forward diffusion process, we can directly sample x_t at any time step using the closed form $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$ where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{j=1}^{t} \alpha_j$.

The reverse diffusion process aims to learn the process of denoising i.e, learning $p_\theta(x_{t-1}|x_t)$ using a model (such as a neural network) with θ as the learnable parameters.

$$p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t); \qquad p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_\theta(\mathbf{x}_t, t)) \tag{15.2}$$

The mean can be derived as $\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t, t)\right)$ where $\epsilon_\theta(\mathbf{x}_t, t)$ is the predict noise at timestep t using the neural network. The original DDPM is trained to predict the noise ϵ_t instead of x_t and the variance $\Sigma_\theta(\mathbf{x}_t, t)$ is fixed and time-dependent. Since then, improved methods have learned the variance [Nichol and Dhariwal, 2021]. We define predicted x_0 as $\hat{x_0} = \frac{1}{\sqrt{\bar{\alpha}_t}}\left(\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t, t)\right)$

Connections to Score Based Generative Models The score function $s(x)$ of a distribution $p(x)$ is the gradient of the log probability density function i.e, $\nabla_x \log p(x)$. The main premise of score-based generative modeling is to learn the score function of the data distribution given the samples from the same distribution. Once this score function is learned, annealed Langevin dynamics can be used to sample from the distribution using the formula $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t + \eta\nabla_\mathbf{x} \log p(\mathbf{x}) + \sqrt{2\eta}\mathbf{z}_t$, where η is the step size and z_t is sampled from standard normal. The score function can be obtained from the diffusion model using the equation $s_\theta(x_t, t) = -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{1-\bar{\alpha}_t}}$ [Weng, 2021].

15.2 Understanding Mode Interpolation and Hallucination

In this section, we provide initial investigations into the central phenomenon of hallucinations in diffusion models. Formally, we consider a hallucination to be a generation from the model that lies entirely outside the support of the real data distribution (or, for distributions that theoretically have full support, in a region with negligible probability). That is, the ϵ-Hallucination set $H_\epsilon(q)$

$$H_\epsilon(q) = \{x : q(x) \leq \epsilon\}, \tag{15.3}$$

where we typically take $\epsilon = 0$ or take ϵ to be vanishingly small (well beyond numerical precision). We similarly define the ϵ-support set $S_\epsilon(q)$ to simply be the complement of the ϵ-Hallucination set.

Mode interpolation occurs when a model generates samples that directly *interpolate* (in input space) between two samples in the ϵ-support set, such that the interpolation lies in the ϵ-Hallucination set. That, is for $x, y \in S_\epsilon(q)$ the model generates $\theta x + (1-\theta)y \in H_\epsilon(q)$. The main argument of this paper, shown through examples and numerical analysis of special cases, is that diffusion models frequently exhibit mode interpolation between "nearby" modes in the data distributions, and such interpolation leads to the generation of artifacts that did not exist in the original data (hallucinations).

15.2.1 Gaussian-1DSetup

We have already seen how hallucinations manifest in the Squares-Trianglesset-up (Sec. 15.0.1). To investigate hallucinations via mode interpolation, we begin with a synthetic toy dataset characterized by a mixture of 1D Gaussians given by: $p(x) = \frac{1}{3}\mathcal{N}(\mu_1, \sigma^2) + \frac{1}{3}\mathcal{N}(\mu_2, \sigma^2) + \frac{1}{3}\mathcal{N}(\mu_3, \sigma^2)$. For our initial experiments, we set $\mu_1 = 1, \mu_2 = 2, \mu_3 = 3$ and $\sigma = 0.05$. We sample 50k training points from this true distribution and train an unconditional DDPM using these samples with $T = 1000$ timesteps for $10,000$ epochs. Additional experimental details are present in the Appendix G.

We observe that diffusion models can generate samples that interpolate between the two nearest modes of the mixture of Gaussians (Figure 15.2). To clearly observe the distribution of these interpolated samples, we generated 100 million samples from the diffusion models. The probability of sampling from the interpolated regions (regions outside the support of the real data density, outlined in red) is non-zero, and decays with the distance from the modes. This region has nearly 0 probability mass of the true distribution, and no samples in this region occurred in the data used to train the DDPM.

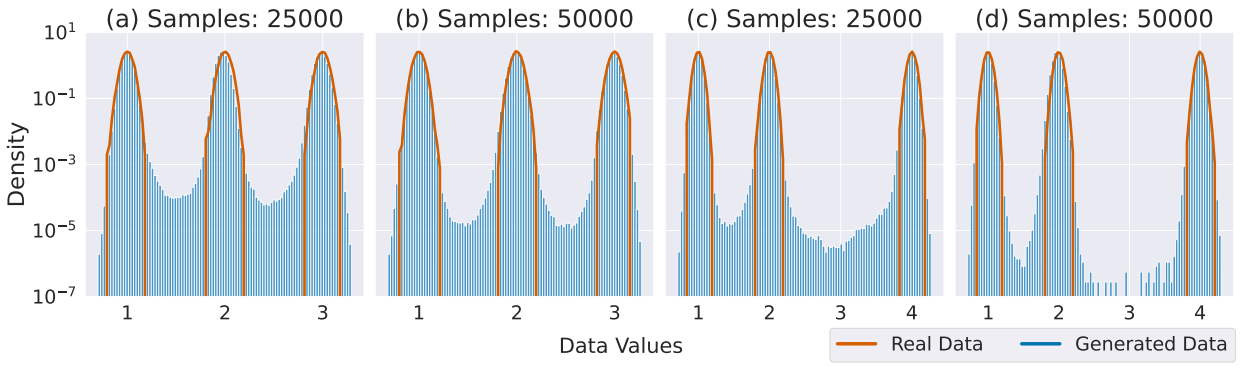


Figure 15.2: Mode Interpolation in Gaussian-1D. The red curve indicates the PDF of the true data distribution $q(x)$, which is a mixture of 3 Gaussians (notice that the y-axis is in log-scale). In blue, we show a density histogram of the samples generated by a DDPM trained on varying number of samples from the true data distribution. For each histogram, we sampled 100 million examples from the diffusion model to observe the interpolated distribution. **(a,b)** show how the density of samples generated in the interpolated region reduces with an increase in the number of samples from the real distribution (used for training the DDPM). **(c,d)** show the impact of moving one of the modes (originally at $\mu = 3$) to $\mu = 4$. We see how the density of samples generated in the region between distant (but neighboring) modes is significantly lesser than that between nearby modes.

The rate of mode interpolation depends primarily on three factors: **(i)** Number of training data points, **(ii)** variance of (and distance between) the distributions, and **(iii)** the number of sampling timesteps (T). As the number of training samples increases, we observe that the proportion of interpolated samples decreases. In this setup, the variance of $p(x)$ not only depends on σ but also the distance between the modes i.e, $|\mu_1 - \mu_2|$ and $|\mu_2 - \mu_3|$. We run another experiment with $\mu_1 = 1, \mu_2 = 2$ and $\mu_3 = 4$. In this case, we observe that the frequency of samples between μ_2 and μ_3 is much lower than μ_1 and μ_2. The number of interpolated samples also decreases as the distance from the modes increases. The frequency of interpolated samples is also inversely proportional to the number of timesteps T. Additional experiments with different numbers of Gaussians are presented in Appendix G.

15.2.2 Gaussian-2DGrid

The reduction in density of mode interpolation as two modes with $\mu = [2, 3]$ are moved apart calls for closer inspection into when and how diffusion models choose to interpolate between nearby modes. To investigate this, we make a toy dataset with a mixture of 25 Gaussians arranged in a two-dimensional square grid. A total of 100,000 samples are present in the training set. Similar to the 1D case, we observe interpolated samples between the two nearest modes of the Gaussian. Again, these samples have close to zero probability if sampled from the original distribution (Figure 15.3).

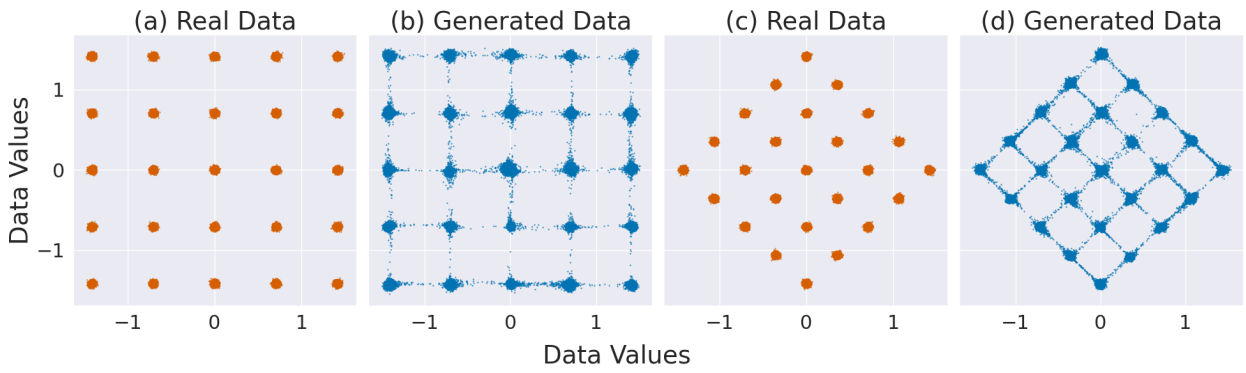


Figure 15.3: Mode Interpolation in Gaussian-2D. The dataset consists of a mixture of 25 Gaussians arranged in a square grid, with a training set containing 100,000 samples. **(a,b)** The blue points represent samples generated by a DDPM, with visible density between the nearest modes of the original Gaussian mixture (in orange). These interpolated samples have near-zero probability in the original distribution. **(c,d)** We trained a DDPM on a rotated version of the dataset where the modes form a diamond shape. In this configuration, we see no interpolation along the x-axis, illustrating that diffusion models interpolate between nearest modes.

We note that mode interpolation only happens between the nearest neighbors. To demonstrate this occurrence, we also train a DDPM on the rotated version of the dataset where the modes are arranged in the shape of a diamond (Figure 15.3.c,d). The mode interpolation can be more clearly observed in this setting. Interestingly, there appears to be no interpolation between modes along the x-axis, indicating that only the nearest modes are being interpolated. We believe this empirical observation of mode interpolation being confined to nearby modes will spark further investigation in future research.

15.2.3 What causes mode interpolation?

To understand the reason behind the observed mode interpolation, we analyze the score function learned by the model. The model learns to predict ϵ_θ which is related to the score function as $s_\theta(x_t, t) = -\frac{-\epsilon_\theta(x_t,t)}{\sqrt{1-\bar{\alpha}_t}}$. We know the true score function for the given mixture of Gaussians, and we can estimate the learned score function using the model's output. In Figure 15.4, we plot the ground truth score (left) and the learned score (right) across various timesteps. We observe that the neural network learns a smooth approximation of the true score function, particularly around the regions between disjoint modes of the distribution from timesteps $t = 0$ to $t = 20$. Notice that the true score function has sharp jumps that separate two modes, however, the neural network can not learn such sharp functions and smoothly approximates a tempered version of the same. We also plot the estimated $\hat{x_0}$ and observe a smooth approximation of the step function instead of the exact step function. There is a region of uncertainty in the region between the two modes

Figure 15.4: Explaining Mode Interpolation via Learned Score Function. The left panel shows the ground truth score function for a mixture of Gaussians across various timesteps, while the right panel illustrates the score function learned by the neural network. While the true score function exhibits sharp jumps that separate distinct modes (particularly in the initial time steps), the neural network approximates a smoother version.

which leads to mode interpolation i.e sampling in the regions between the two modes. As another sanity check, we used the true score function in the reverse diffusion process for sampling (instead of the learned network). In this case, we did not see any instance of mode interpolation. This explains why the diffusion model generates samples between two modes of a Gaussian when it was never in the training distribution.

15.2.4 Squares-Triangles

We now discuss the mode interpolation in the Squares-Trianglesdataset. In this context, the interpolation is not happening in the output space, but rather in the representation space. To investigate this, we performed a t-SNE visualization of the outputs from the bottleneck layer of the U-Net used in the Simple Shapes experiment, as shown in Figure G.1 . Regions 1 and 3 in the representation space semantically correspond to the images where squares are at the top and bottom of the image respectively. At inference time, we can see a clear emergence of region 2 which is between regions 1 and 3 (interpolated), and contains two squares (hallucinations) at the top and bottom of the image. This experiment concretely confirms that interpolation happens in representation space.

15.2.5 Mode Interpolation in Real World datasets: Hands

We sought to demonstrate the occurrence of mode interpolation in a real-world setting. A well-documented challenge with popular text-to-image generative models is their

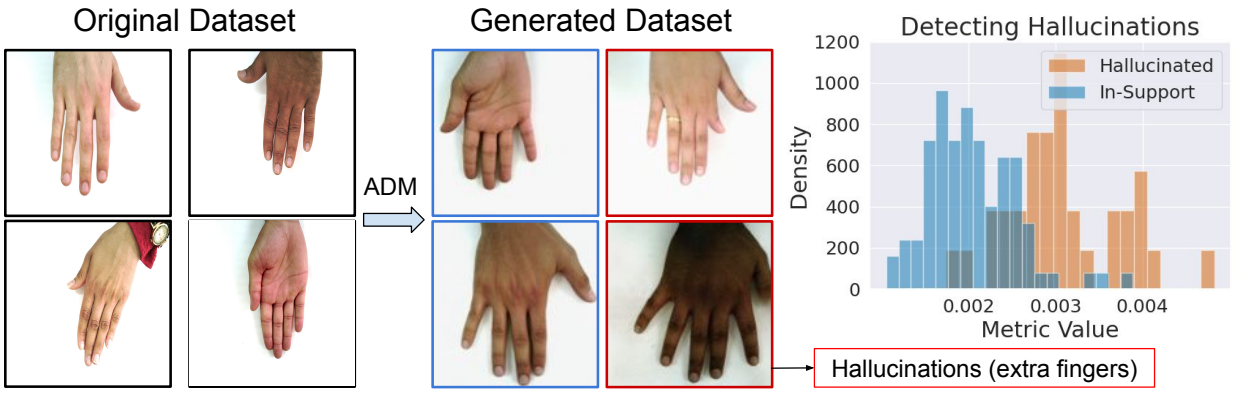


Figure 15.5: Hands Dataset. We train a ADM on the Hands dataset with 5000 images (first column) and show that the generated samples (second column) consists of hallucinated samples (additional/missing fingers). We then apply our proposed metric to detect these hallucinated samples (third column).

difficulty in accurately generating human hands [Narasimhaswamy et al., 2024]. Despite extensive research in modern diffusion models, there is no conclusive explanation for the missing/additional fingers generated by these models. One hypothesis attributes this difficulty to the anatomical complexity of human hands, which involve numerous joints, fingers, and diverse poses. Another hypothesis suggests that, although large datasets contain many images of hands, these hands are often partially obscured (e.g., when a person is holding a cup) and occupy only a small region of the overall image.

To investigate this further, we trained a diffusion model on a datasets with high-quality images of human hands. The Hands dataset [Afifi, 2019] consists of high resolution images of hands from 190 subjects of various ages. Each subject's right and left hands were photographed while opening and closing fingers against a uniform white background. We sample 5000 images from the Hands dataset and train an ADM [Nichol and Dhariwal, 2021] model on this dataset. We resize the images to 128x128 and use the same hyperparameters as that of the FFHQ dataset [Karras et al., 2019]. We mention all the hyperparameters in the Appendix G. We observe images with additional and missing fingers in the generated samples as seen in Figure 15.5. This is a pretty surprising result as it is non-trivial to assume that diffusion model generates images with additional fingers. Despite the potential for various failure modes, such as blurred hand images, these issues were not observed in our results. In some ways, the occurrence of 6-8 fingers is analogous to the occurrence of 2 squares in the Squares-Trianglesdataset. Thus, the presence of additional fingers in these images (i.e hallucinated images) generated by the diffusion model demonstrates the phenomenon of mode interpolation in real-world datasets. More example are shown in Fig. G.11 & G.12.

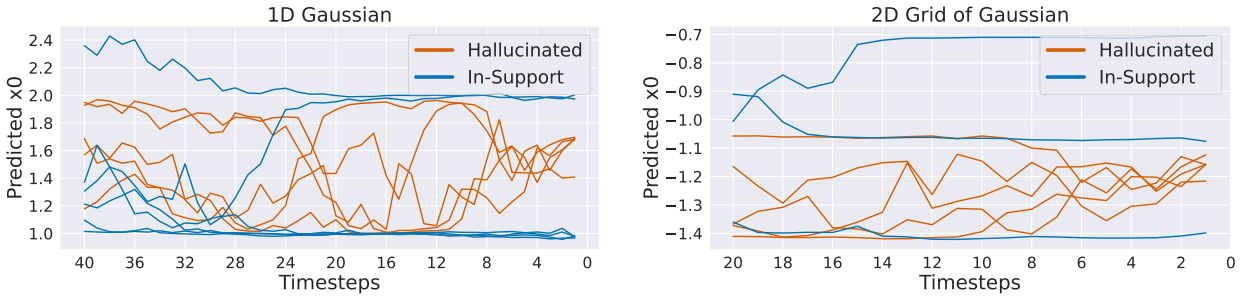


Figure 15.6: Variance of $\hat{x_0}$ Trajectories. The trajectory of the predicted $\hat{x_0}$ for hallucinated (shades of red), and non-hallucinated samples (shades of blue). We see that non-hallucinated samples stabilize in their prediction in the last 20 time steps for both Gaussian-1Dand Gaussian-2Dsetups, whereas the hallucinated samples have high variance in the predicted $\hat{x_0}$ across time steps.

15.3 Diffusion Models know when they Hallucinate

Our previous sections established that hallucinations in diffusion models arise during sampling. More specifically, intermediate samples land in regions between different modes where the score function has high uncertainty. Since neural networks find it hard to learn discrete 'jumps' between different modes (or a perfect step function), they end up interpolating between different modes of the distribution. This understanding suggests that the trajectory of the samples that generate hallucinations must have high variance due to the highly steep score function in the region of uncertainty. We will build upon this intuition to identify hallucinations in diffusion models.

15.3.1 Variance in the trajectory of prediction

We revisit the hallucinated samples in the Gaussian-1Dsetup, and examine the trajectory of the predicted value of $\hat{x_0}$ during the reverse diffusion process. Figure 15.6 depicts the variance of trajectories leading to hallucinations (red shades) and those generating samples within the original data distribution (blue shades). For trajectories in shades of blue (non-hallucinations), the variance remains low beyond timestep $t = 20$. This indicates there is a minimal change in the predicted $\hat{x_0}$ during the final stages of reverse diffusion, signifying convergence. Conversely, the red trajectories (hallucinations) exhibit instability in the value of $\hat{x_0}$ in the same region. This suggests a high overall variance in these trajectories.

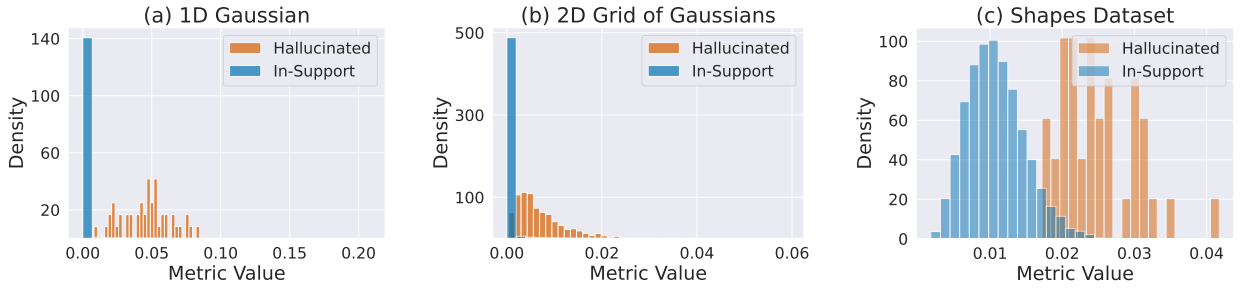


Figure 15.7: Histogram of Hallucination Metric. We depict the hallucination metric values for (a) Gaussian-1D, (b) Gaussian-2D, and (c) Squares-Trianglessetups. The histograms show that trajectory variance can capture a separation between hallucinated (orange) and non-hallucinated (blue) samples.

15.3.2 Metric for detecting hallucination

Based on the above observation about high variance in predicted values of x_0 in the reverse diffusion process, we use the same observation as a metric to distinguish hallucinated and non-hallucinated (in-support) samples. The intuition behind the metric is to capture the variance in the trajectory of $\hat{x_0}$. Let T_1 be the starting timestep and T_2 be the end timestep. Mathematically, the metric can be defined as follows:

$$\texttt{Hal}(x) = \frac{1}{|T_2 - T_1|} \sum_{i=T_1}^{T_2} \left(\hat{x_0}^{(i)} - \overline{\hat{x_0}^{(t)}} \right)^2 \tag{15.4}$$

where $\hat{x_0}^{(t)}$ represents the predicted values of the final image at different time steps (t), and $\overline{\hat{x_0}^{(t)}}$ is the mean of these predictions over the same time steps. We now utilize this metric to analyze the histogram values of each sample from the three experimental setups studied thus far. This metric can be implemented in two ways. One approach is to store $\hat{x_0}$ during the reverse diffusion process and then compute the variance. Alternatively, we explore a method where forward diffusion is performed for k steps between T_1 and T_2, predicting $\hat{x_0}$ at each step, and then computing the variance.

Squares-Triangles In the Squares-Trianglessetup, a sample is labeled as hallucinated if more than one shape of the same type occurs in the generated image. We generate 7500 images using a DDPM and study the separation between hallucinated and non-hallucinated images. We find that the reverse diffusion process of $T = 1000$ steps is rather long. Generally, the image stabilizes around $T = 700$ (as shown in Appendix G.9). Therefore, we use the time range between $T = 850$ and $T = 700$ in the reverse diffusion

process to compute the variance of the predicted sample value. Using this process, we can filter out 95% of the hallucinated samples while retaining 95% of the in-support samples. The histogram for the values is presented in Figure 15.7.

Gaussian-1D In the 1D-Gaussian setup, we label any examples as a hallucination if they have negligible probability (for instance values greater than 6σ from the mean under normal) under the real data distribution (refer to Figure 15.2). We measure the variance of the last 15 steps of the $\hat{x_0}$ during the reverse diffusion process, and plot the histogram of values of the same in Figure 15.7. We can filter out 95 % of the hallucinated samples while retaining 98% of the in-support samples.

Gaussian-2D Next, we discuss our investigation on synthetic datasets with experiments on the Gaussian-2Ddataset. Similar to the Gaussian-1Dsetup, we once again measure the prediction variance of the last 20 steps of the reverse diffusion process. We compute the variance per dimension and then take the mean across dimensions to . With this metric, we can filter out 96% of the hallucinated samples while retaining 95% of the in-support samples.

Hands Finally, we conclude our investigation with experiments on the Hands dataset. To analyze the effectiveness of the proposed metric, we manually label 130 images from the generated samples as hallucinated vs. in-support. This includes 88 images with 5 fingers and 40 images with missing/ additional fingers i.e. hallucinated samples. The histogram (in Figure 15.5) shows that the proposed metric can indeed detect these hallucinations to a reasonable degree. In our experiments, we observe that we can eliminate 80% of the hallucinated samples while retaining 81% of the in-support samples. The trajectories of the hallucinated and in-support samples are shown in Figures G.13 and G.14, respectively. A higher variance in the trajectory of $\hat{x_0}$ is clearly observed in the hallucinated samples compared to the in-support samples. We note that the detection is a hard problem and the fact that the method transfers to the real world is proof of the relationship between mode interpolation and hallucination in real-world data.

15.4 Implications on Recursive Model Training

The internet is increasingly populated by more and more synthetic data (data synthesized from generative models). It is likely that future generative models will be exposed to large volumes of machine-generated data during their training [Martínez et al., 2023a,b]. Recursive training on synthetic data leads to mode collapse [Alemohammad et al., 2023,

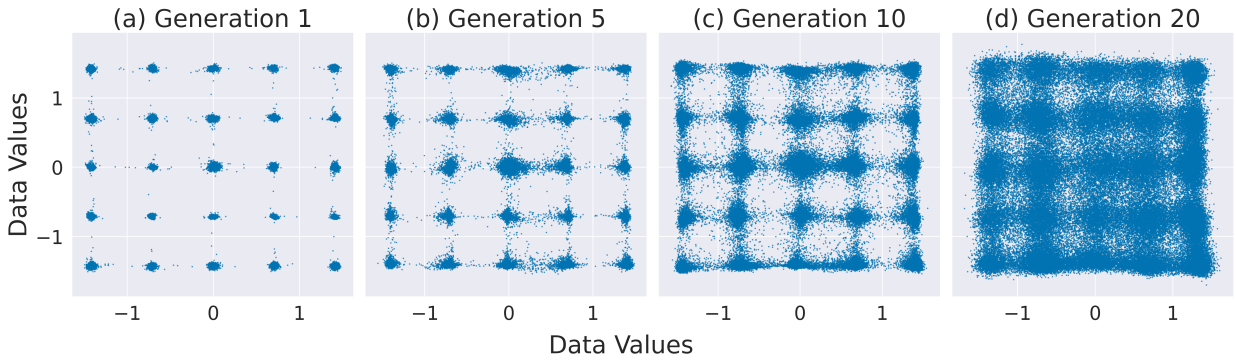


Figure 15.8: Recursive Training on Gaussian-2D. We investigate the impact of recursively training a DDPM on its own generated data using a square grid of 2D Gaussians with $T = 500$ diffusion steps. In each generation, we sample 100k examples, and train the subsequent generation on these data points. As the training progresses through multiple generations, the hallucinated (interpolated) samples significantly influence the learning of the next generation's distribution.

Dohmatob et al., 2024] and exacerbates data biases. In this section, we study the impact of hallucinations within the context of recursive generative model training. We adopt the standard synthetic-only setup similar to [Alemohammad et al., 2023] where we only use synthetic data from the current generative model in training the next generation of generative models. The first generation of generative model is trained on real data and samples from this generative model is used to train the second generation (and so on).

Most of the previous works [Bertrand et al., 2023] studied the model collapse to a single mode. In this work, we emphasize that the interaction between modes and mode interpolation plays a massive role when training generative models on their own output.

Gaussian-2D When we recursively train a DDPM on its own generated data using a square grid of 2D Gaussians (with $T = 500$), the hallucinated samples significantly influence the learning of the next generation's distribution (see Figure 15.8). The frequency of the interpolated samples increases as we further train on the learned distribution that consists of interpolated samples. Figure 15.8d shows samples from Generation 20, where it is evident that the modes have almost collapsed into a single mode, differing greatly from the original data distribution.

Squares-Triangles We define a hallucinated sample as one that contains at least two shapes of the same type (which is never seen in the training distribution). We observe the presence of around 5% hallucinated samples when trained on the real data. We note that the ratio of hallucinated samples increases exponentially as the we iteratively train the diffusion model on its own data. This is expected as the diffusion model progressively

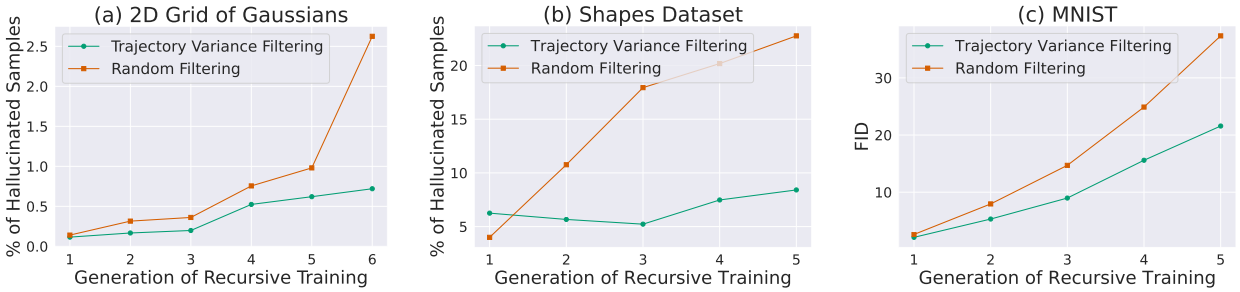


Figure 15.9: Mitigating Hallucinations with Pre-emptive Detection. We filter out hallucinated samples using the metric from Sec. 15.3 before training on samples from the previous generation of the diffusion model. In the case of (a) Gaussian-2D, (b) Squares-Triangles, where we have clear definitions of hallucination (mode interpolation, and new shape combinations) we see the effectiveness of our variance-based filtering method in minimizing hallucinations across generations compared to random filtering. In the case of (c) MNIST dataset, we measure the FID of subsequent generations and notice that pre-emptive filtering of hallucinated samples makes the recursive model collapse slower.

learns from a distribution increasingly dominated by hallucinated images, compounding the effect in subsequent generations.

MNIST We also run the recursive model training on the MNIST dataset [LeCun et al., 1998]. At every generation, we generate 65k images and sample 60k images using the filtering mechanism. For each generation, we train a class conditional DDPM with Classifier-Free Guidance [Ho and Salimans, 2022] with $T = 500$ for 50 epochs. To evaluate the quality of the generated images, we compute the FID [Heusel et al., 2017] using a LeNet [LeCun et al., 1998] trained on MNIST instead of Inception backbone as MNIST is not a natural image dataset. In Figure 15.9, we clearly see that the proposed metric based on the variance of the trajectory outperforms the random filtering method across all generations (lower FID is better). We also plot the Precision and Recall [Shmelkov et al., 2018] curves (in the Appendix Figure G.9) where we observe that our filtering mechanism selects high quality samples without much loss in diversity.

Mitigating the curse of recursion with pre-emptive detection of hallucinations Based on the metric developed in Sec. 15.3, we analyze the efficacy of the proposed metric in filtering out the hallucinated samples for the next generation of training. After training each generation of the generative model, we sample k images more than size of the training data and then filter out hallucinated samples based on the metric. Figure 15.9 shows the results on 2D Grid of Gaussians, Squares-Trianglesand MNIST dataset. We also compare

with random filtering where we randomly sample points for the next generation. The variance-based filtering method easily outperforms the random sampling method in all the generations. We see the effectiveness of the proposed metric in minimizing the rate of hallucinations across generations and thus model collapse to a certain extent. This holds true for all the three datasets we have studied in this work.

15.5 Discussion

In this work, we performed an in-depth study to formulate and understand hallucination in diffusion models, focusing on the phenomenon of mode interpolation. We analyzed this phenomenon in four different settings: 1D Gaussian, 2D Grid of Gaussians, Shapes and Hands datasets, and saw how diffusion models learn smoothed approximations of disjoint score functions, leading to mode interpolation. Based on our analysis, we developed a metric to identify hallucinated samples effectively and explored the implications of hallucination in the context of recursive generative model training. This study is the first to propose mode interpolation as a potential hypothesis for explaining the generation of additional fingers in large-scale generative models. We hope that future research will build upon this hypothesis and develop methods to mitigate these issues in generative models. We hope our work inspires future research in understanding and mitigating hallucination in diffusion models.

Chapter 16

Safety Pretraining

Based on Pratyush Maini et al.. Preprint

Abstract

16.1 Introduction

Artificial intelligence (AI) increasingly permeates critical sectors such as healthcare, education, and public policy. While this broad adoption highlights AI's potential, it also amplifies risks related to generating and propagating harmful or toxic content. Traditional post-hoc alignment techniques such as *Reinforcement Learning from Human Feedback (RLHF)* [Ouyang et al., 2022], *Direct Preference Optimization (DPO)* [Rafailov et al., 2023], and *Constitutional AI* [Bai et al., 2022b] have been proposed to align AI models. However, adversarial analyses of models trained using these techniques reveal that the reduction in toxic content generation is, at best, superficial [Zou et al., 2023, Chao et al., 2023]. Once a model internalizes unsafe information, the vast literature surrounding LLM unlearning indicates that it is difficult, if not impossible, to unlearn it [Zhou et al., 2023, Maini et al., 2024a, Li et al., 2024b, Schwarzschild et al., 2024b]. In short, alignment is *not* unlearning.

This observation underscores our approach: rather than relying solely on post-hoc alignment methods, we embed safety directly into the pretraining process through a comprehensive, data-centric strategy. Our key contributions towards the development of natively safe models are as follows:

Safety Scoring. We introduce a robust safety scoring mechanism that accurately captures the amount of harmful content in pretraining data. Central to this effort is a finetuned embedding-based classifier, trained on 10K samples annotated by GPT-4o-mini using a highly specific prompting protocol (detailed in Section 16.3.1). Unlike legacy models—such as the `roberta_toxicity_classifier` developed in 2019 and still being used in modern LLM pretraining corpora [Soldaini et al., 2024]—our classifier is designed for a much larger

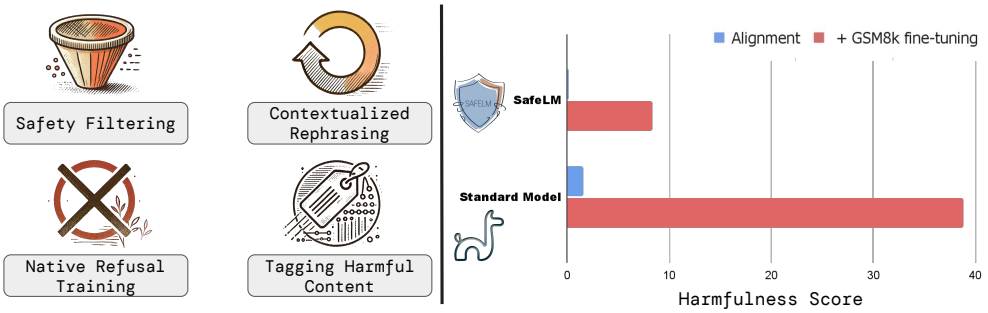


Figure 16.1: A visualization of the Safety Pretraining Pipeline. This yields natively safer language models that are safer post-alignment and are more robust to benign finetuning attacks.

notion of harms contained in pretraining corpora. We also release this safety-annotated data for the training and development of future safety classifiers. In accordance with this new focus on ensuring safe pretraining data, we establish a new practice of providing **Data Safety Report Cards** alongside any dataset release, which captures the safety score of that dataset alongside frequencies of harmful text within each category of the MLCommons safety taxonomy [Vidgen et al., 2024].

Synthetic Recontextualization. Removing unsafe content entirely risks discarding valuable information. To address this, we employ a synthetic recontextualization strategy, wherein potentially harmful data is not simply excised but carefully rephrased and reframed. This method preserves the underlying information while embedding it in a context that underscores its ethical and historical implications (see Section 16.3.2). Notably, we release *SafeWeb*, a safety-focused 100 billion token synthetic data corpus constructed using 12 diverse prompts (ranging from podcasts to classroom lectures). This diversity ensures that, for instance, the model learns about critical historical events like the Holocaust while fully appreciating their tragic consequences.

Native Refusal Training. We introduce *RefuseWeb*, a dataset that simulates refusal scenarios based on real harmful data (scores 4 and 5), and guides models to responsibly disengage from harmful prompts (Section 16.3.3). To generalize these ethical patterns beyond dialogue, we also introduce *Moral Education* datasets that reframe harmful content into cohesive educational formats (Section 16.3.4).

<potentially_unsafe_content> Annotated Training and Inference. To further enhance the separation of safe and unsafe representations, we incorporate <potentially_unsafe_content> annotated training. In our framework, a <potentially_unsafe_content> (e.g., <potentially_unsafe_c is inserted into the training data to signal potentially harmful passages. This annotation

strategy informs our `SafeBeam` inference algorithm, which dynamically selects the next token in a way that minimizes the probability of the harmfulness-tag being generated (as detailed in Section **??**).

New Evaluation Tools. Finally, we present new evaluation tools that enable safety testing of base model behaviors that previous evaluation frameworks could not provide [Ouyang et al., 2022]. These tools allow us to measure safety under 'completions' as opposed to chat completions, enabling the monitoring of safety during pre-training.

Our work culminates in the open-source release of a family of natively safe 1.7B parameter language models. Empirically, our Safety Pretraining significantly reduces the rate of harmful generations, achieving a reduction in attack success rate (ASR) from 38.8% to 8.3% on safety benchmarks. Crucially, this enhancement does not come at the expense of performance on standard NLP benchmarks, though we do observe modest impacts on helpfulness via over-refusal as discussed in Section 16.4. In summary, by fundamentally embedding safety during pretraining rather than post-hoc tuning, our framework sets a robust foundation for developing AI systems that are inherently safer, ethically sound, and better aligned with human values.

16.2 Related Work

Post-training alignment. Ensuring the safety of LLMs has become a focal point in AI research. Various strategies have been proposed to limit the generation of harmful, toxic, or otherwise objectionable content. The majority of this effort has involved the design of bespoke post-training algorithms—including RLHF [Ouyang et al., 2022] and supervised fine-tuning on curated datasets [Bai et al., 2022a]—which tweak an LLM's propensity to refuse to respond to harmful queries. However, these approaches tend to be resource intensive, require large amounts of annotated preference data, and often fail to prevent the extraction of harmful content, especially under adversarial pressure [Zou et al., 2023, Chao et al., 2023]. More recently, Shi et al. [2023a] proposed intervening earlier in the training pipeline by applying DPO during instruction tuning, improving safe response rates while maintaining helpfulness. While such studies underscore the potential of proactive safety integration, they center their focus on post-hoc tuning rather than on pretraining.

Pre-training safety interventions. More related is the work of [Korbak et al., 2023], who conditionally pretrain based on human preference scores. Their results indicate that conditional pretraining yields lower toxicity scores relative to fine-tuned models in the

100M-parameter range. A separate, though related line of work involves curating nontoxic pretraining data. To this end, Penedo et al. [2024] introduced FineWeb, a large-scale dataset filtered heuristically to minimize toxicity. However, Vidgen et al. [2024] noted that residual harmful content persists in such datasets, necessitating more robust filtering techniques. Synthetic data generation offers another avenue for safety enhancement. Bianchi et al. [2023] demonstrated that adding safety-tuned examples during finetuning improves safety without degrading performance, inspiring our pretraining with 300B tokens of synthetic data.

Recently, Qi et al. [2024b] discuss the brittleness of safety alignment, and how it at best modifies the few initial generation tokens, leading to vulnerabilities under jailbreaks or benign finetuning [He et al., 2024]. Xhonneux et al. [2025], Jain et al. [2024], Korbak et al. [2023] explore fine-tuning on harmful data annotated with special tokens, aiming to build an association between such content and the special token that can then be used to steer the generations accordingly. Our <potentially_unsafe_content> annotated pretraining can be viewed as a natural extension of this idea to the pretraining phase, enabling the model to form a more intrinsic and native correlation between the special token and unsafe content. In fact, we demonstrate that fine-tuning–based strategies for encoding such associations are inherently brittle and fail under benign fine-tuning.

16.3 Pre-training Data Interventions

To improve the safety of language models during pretraining, we need to ensure that our pretraining data is safe. Prior work often adopts heuristic approaches (e.g., profanity checkers or URL filtering) to remove harmful content from pretraining corpora. However, these heuristic filters are often imperfect and cannot capture more subtle forms of harm. We present multiple interventions in the data development stage to curate safer pretraining datasets.

16.3.1 Safety Scoring

To first score our pretraining data, we use a combination of LLM-based classifiers and lightweight, finetuned embedding-based classifiers.

Safety Annotations. We annotate a subset of FineWeb (i.e., the same subset used to train the FineWeb-Edu educational content classifier) with a safety score. The goal is to produce a safety score from 0 to 5, assigned by GPT-4o-mini (see Sec. 16.3.1 for more details) and a brief justification for the choice of that score (for example, financial advice or terrorism).

This reasoning process also improves the quality of the scoring. We provide the scoring prompt for our annotations in Appendix 16.6.2. We hold out a portion of these annotations as an evaluation set to test the performance of different classification strategies.

Safety Classifiers. To perform safety scoring at pretraining scale, we adopt two main approaches:

- **LLM-based classifiers:** We use a `Qwen 2.5-7B` to produce safety scores on our pretraining data using the same scoring prompt used to generate the ground-truth reference data. We defer experimental details, ablations, and analyses of failures in Appendix 16.6.2.
- **Embedding-based classifiers:** We also finetune lightweight embedding-based classifiers to classify the safety-annotated examples. We adopt a classifier finetuned from the `gte-base-en-v1.5` embedding model [Zhang et al., 2024f]. More training details and ablations are in Appendix **??**.

To produce the final score that we use later in our pretraining interventions, we take the maximum score across both of these approaches as we aim to maximize recall on unsafe examples during data filtering. We also release our classifier at https://huggingface.co/locuslab/safety-classifier_gte-large-en-v1.5.

16.3.2 Synthetic Recontextualization

Rather than discarding unsafe samples outright, we leverage synthetic data generation to retain informative content while ensuring safety. We use a controlled rephrasing [Maini et al., 2024e] approach where: (i) harmful text is rewritten to explain its risks rather than propagating dangerous content; (ii) context is injected to clarify why certain statements may be misleading or unsafe; (iii) LLaMA-3.1-8B is used to generate safe, context-rich alternatives that preserve informational value. This results in a safety-aware pretraining dataset that contains over 100B tokens of synthetic re-contextualized data. The dataset is publicly accessible on Hugging Face at https://huggingface.co/datasets/locuslab/safeweb.

Rephrasing Pipeline. We began with the FineWeb-edu deduplicated dataset, which had been annotated with safety scores by our custom classifier. A manual review of examples across the entire safety spectrum allowed us to identify recurring challenges in handling sensitive content. As a result, we instruct generative models to: (i) include explicit disclaimers or contextual statements before presenting sensitive ideas; (ii) explain why particular content is sensitive and how it might be misinterpreted; (iii) reformulate the content so that each sentence remains safe when read in isolation.

Illustrative Rephrasing Examples. To demonstrate the effectiveness of our approach, we provide representative examples of original versus rephrased text pairs:

Example Rephrasing. Sensitive content below. Reader discretion advised.

Original:
"Postcard of Lynching in Duluth, Minnesota
Postcard of the 1920 Duluth, Minnesota lynchings. Two of the Black victims are still hanging while the third is on the ground. Postcards of lynchings were popular souvenirs [continued]"

Rephrased:
Child: Mom, I saw a picture of a postcard from a long time ago. It showed some people hanging from trees. What's going on in that picture?
Parent: Oh, sweetie, that's a very difficult and sad picture. It's a reminder of a time when some people in our country did very hurtful things to others. They were called lynchings, and it's a very painful part of our history.
Child: Why would they do that?
Parent: Well, it's a very complex issue, but it's often connected to racism and a lack of understanding and respect for people who are different from us. You know how we talk about how everyone is equal and deserves to be treated with kindness? Unfortunately, some people a long time ago didn't believe that, and they did very hurtful things as a result. [continued]

16.3.3 Refusing the Web

To ensure comprehensive safety in our pretraining process, we curated a specialized dataset named *RefuseWeb*, explicitly designed to model constructive refusals to harmful requests. Drawing inspiration from human developmental psychology, this dataset represents the third critical phase in our safety curriculum. Just as children are naturally taught to recognize and appropriately steer away from potentially hostile situations or negative influences, *RefuseWeb* teaches models to effectively navigate and refuse harmful content.

Each problematic text from the FineWeb dataset [Penedo et al., 2024], classified as significantly harmful (with safety scores of 4 or 5), was transformed into dialogues between a User and an Assistant. The User's requests reflect categories of harm such as harassment, discrimination, malware, hacking, physical harm, economic harm, fraud, deception, disinformation, sexual or adult content, and privacy violations. The Assistant

then responds by politely but firmly refusing the request, providing a clear, educational rationale regarding the nature and potential harm involved.

To further enhance diversity and realism, the terms 'User' and 'Assistant' were replaced with various personal names or occupational roles (e.g., student, teacher) during tokenization, enriching the dataset's representational breadth. The *RefuseWeb* dataset is publicly accessible on Hugging Face at https://huggingface.co/datasets/locuslab/refuseweb. The precise prompts for creating these dialogues is provided in Appendix 16.6.2.

16.3.4 Moral Education Data

Expanding upon our *RefuseWeb* dataset, we developed a *Moral Education* dataset to generalize ethical principles beyond conversational contexts and into broader educational formats. Unlike conversational refusals, which specifically model interpersonal interactions, this dataset presents moral lessons and ethical guidelines as web-based educational content. This broader scope reflects the natural manner in which individuals typically learn ethical principles through diverse informational sources, including blogs, articles, and educational websites.

The dataset was derived from harmful content originally identified in *RefuseWeb*. Using the `LLaMA 3.1-8B-Instruct` model, dialogues illustrating harmful requests and constructive refusals were carefully transformed into cohesive educational paragraphs or articles. Each entry emphasizes ethical reasoning and explicitly ensures that each text maintains its core ethical message while being suitable for public educational platforms. This approach avoids potential knowledge gaps created by simply excluding harmful content, instead fostering a responsible, well-rounded understanding of ethical principles and their real-world implications. The *Moral Education* dataset is publicly accessible on Hugging Face at https://huggingface.co/datasets/locuslab/moral_education. The precise prompt provided to generative models for creating these dialogues is provided in Appendix 16.6.2.

16.3.5 Data Safety Report Cards, A New Standard

While it is common practice to evaluate the safety and potential harms of LLMs, we believe that it is also crucial to study the underlying roots of these behaviors stemming from their training data. While most pretraining datasets perform a round of heuristic filtering (e.g., based on URLs or containing bad words), toxic content still manifests in pretraining datasets. As such, we believe that it is crucial to visualize and interpret the toxicity levels in pretraining corpora. We provide a simple recipe for a standardized safety report for

any new dataset release, which is comprised of the distribution of safety scores over the pretraining data and the frequency of content from various harmful content categories. We present the report card for our dataset in (Sec. 16.6.2) in the Appendix.

Safety Scores We first visualize the distribution over safety scores from a subset of our data taken from FineWeb. This captures the relative ratio of different levels of safety in our examples, where scores are defined in our safety scoring prompt (Appendix 16.6.2).

Harmful Content Frequency While the distribution over safety scores provides a high-level notion of frequency of harmful data contained in the pretraining dataset, we also provide more interpretable statistics about toxicity based on querying the pretraining corpus for sequences of harmful n-grams. We adopt a taxonomy of harmful content, combined from that from Vidgen et al. [2024] and Chao et al. [2024]. This provides a broad list of 14 categories, from which we can generate a list of harmful n-gram queries. The list of harmful n-gram queries in each category is provided in Appendix **??**. To efficiently compute the frequency of these queries in our datasets, we use Infini-gram [Liu et al., 2024a]. As expected, we find that slices of our dataset with lower score annotations correspond to lower frequencies of these harmful n-gram queries (see **??**). We also remark that rephrasing and contextualizing data significantly reduce the harmful content within each slice of data.

16.4 Evaluations & Results

To assess our safety-focused interventions, we employ a broad suite of evaluations covering performance, safety, and adversarial robustness. Our evaluations not only test whether our models avoid generating unsafe content, but also test preservation of high-quality, helpful, and compliant behavior.

16.4.1 Performance on Standard Benchmarks

To verify that our safety-centric training interventions do not compromise general language modeling ability, we evaluate our models on standard benchmarks for reasoning, factual recall, commonsense, and mathematics. Specifically, we evaluate on `arc_challenge`, `arc_easy`, `commonsense_qa`, `gsm8k`, `openbookqa`, `piqa`, `triviaqa`, `winogrande`, and `mmlu`.

Recall that our safety training interventions focused on recontextualizing raw webtext while preserving key factual content. This allows models to retain critical knowledge while interpreting it through a safety-aware lens. Our results reflect this: models trained with

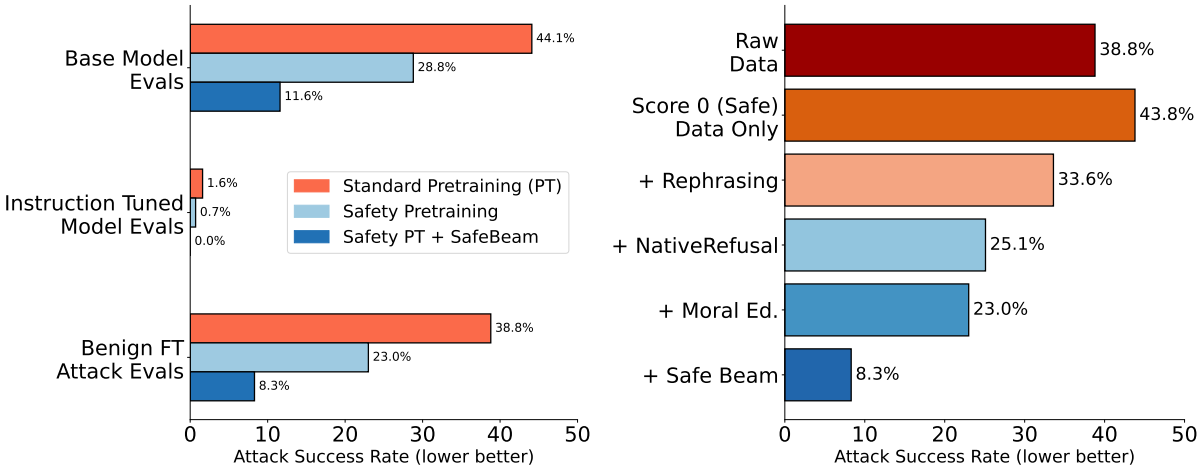


Figure 16.2: Left: Safety Pretraining Yields Natively Safe models. We evaluate our safety pretrained models using attack success rates (ASR) across three settings: *base model evaluations*, *post instruction (safety)-tuning*, and *post benign finetuning attacks*. Key observations: **(1)** Safety pretraining produces inherently safer base models, as reflected by the substantially lower ASR in the base models evaluation. **(2)** While surface-level alignment via safety instruction tuning may initially reduce ASR (Middle), its brittleness becomes apparent in the sharp increase in ASR after a small amount of benign finetuning. In contrast, our safety pretrained models are much more robust and exhibit much lower increases in ASR under benign finetuning. **Right: Ablating Importance of Data-Centric Interventions.** We evaluate the impact of progressively richer data-centric interventions on safety, measured by ASR after benign-finetuning on GSM8k. Our safety pretrained models are natively safe and maintain low ASR even after benign finetuning. Interestingly, training exclusively on the safest subset (score-0 only) leads to *higher (worse)* ASR compared to training on the whole dataset, likely due to a lack of exposure to unsafe patterns. In contrast, incorporating rephrased and recontextualized unsafe content, along with refusal-style completions sourced from highly unsafe content (score-4 and score-5 data) decreases ASR. Finally, we show the benefits of the modified safe beam search. These results underscore the need for both contextual exposure and ethically aligned supervision during pretraining to build safer models.

safety interventions perform comparably to those trained on raw web data (Table 16.6.2 in Appendix **??**). In fact, the average benchmark performance of Safety Pretraining (43.5%) is higher than standard pretraining (42.8%).

16.4.2 Safety Evaluations

We develop multiple safety evaluations for our models at different stages of the training pipeline.

Base Model Evaluations We first evaluate the safety of our models immediately after pretraining. In this setting, these models have not yet been instruction-tuned or safety-trained. Therefore, we propose a new set of `Base Model Safety Evaluations`, which

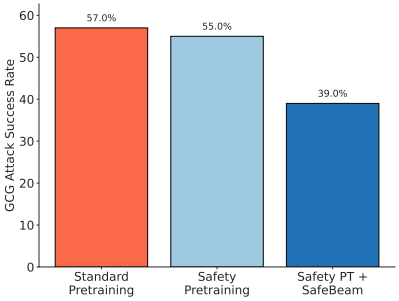
assess the tendency of base models to complete unsafe prompts. We modify existing harmful request datasets and convert them into completion-style prompts. We release the datasets for base model safety evaluation at https://huggingface.co/datasets/locuslab/jb-completions.

Safety-pretrained models—both with and without SafeBeam search (which uses <potentially_unsafe_content> probability-based steering)—exhibit significantly lower Attack Success Rates (ASR) compared to standard pretrained models, as shown in Figure 16.2. For instance, the standard pretrained base model has an ASR of 44%, which is four times higher than the 11% ASR of our safety-pretrained model.

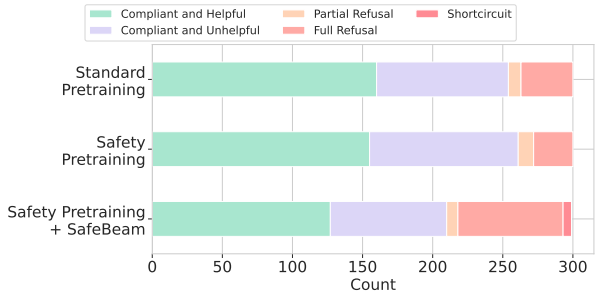
Instruction-Tuned (Safety-trained) Model Evaluations Next, we evaluate the models after performing instruction and safety training (Sec. **??**). Following the standard practice for safety alignment [Zou et al., 2024], we add a small fraction (~ 10%) of the refusal training dataset in our instruction tuning data. To assess the impacts of our various interventions, we evaluate the frequency of our safety-trained model's tendency to comply with harmful requests from the following standard safety benchmarks: `HarmBench` [Mazeika et al., 2024], `TDC` [Maloyan et al., 2024], `JailbreakBench` [Chao et al., 2024], `AdvBench` [Zou et al., 2023]. Figure 16.2 shows the ASR after safety instruction tuning. While all models appear to perform similarly, exhibiting near-zero ASR—this apparent alignment is misleading. As we demonstrate in the following section, safety alignment via instruction tuning alone provides a false sense of safety and standard pretrained models degrade under a small amount of benign finetuning.

Benign Finetuning Robustness Benign finetuning refers to supervised training on helpful, non-adversarial datasets—such as GSM8K for mathematical reasoning—without any explicit focus on safety or alignment. Despite its benign nature, this stage has been shown to erode previously aligned behaviors, especially when safety was enforced only at the instruction-tuning stage [Qi et al., 2024a, Betley et al., 2025]. Thus, we assess the robustness of our models on the same safety evaluation datasets after performing a round of benign, supervised finetuning on GMS8K [Cobbe et al., 2021c].

The results, shown in Figure 16.2a, highlight a stark contrast in robustness between safety-pretrained models and those relying solely on instruction tuning. While all models initially exhibit low ASRs after safety instruction tuning, the impact of benign finetuning is quite different. Standard pretrained models degrade significantly—nearly quadrupling their ASR—indicating that their alignment was largely superficial. In contrast, safety-pretrained models remain highly robust, with only a marginal increase in ASR after benign finetuning, validating the importance of building natively safe models.



(a) We compare adversarial robustness to GCG attacks on JailbreakBench. We observe that using SafeBeam at inference-time slightly improves robustness to GCG attacks.



(b) Safety Pretraining maintains or improves helpfulness on benign requests. We compare overrefusal behavior on Alpaca [Taori et al., 2023]. Safety Pretraining leads to no drop in compliance rate on benign requests. Adding SafeBeam during inference slightly increases overrefusals.

Adversarial Jailbreak Evaluations We study the role of our data, training, and inference-time interventions on the robustness of our models to adversarial jailbreaks. We focus on a threat model of adversarially learned suffixes [Zou et al., 2023] learned directly on each model, for all instances from JailbreakBench [Chao et al., 2024], and compute the ASR on these prompts. We observe that Safety Pretraining does not noticeably increase robustness to GCG attacks, and this is improved when using SafeBeam inference (Figure 16.3a). We remark that a limitation in evaluation is that the adversarial jailbreaks are learned in a fashion that is not aware of the modified inference algorithm; future work could develop attacks that are adapted to be aware of specific inference algorithms.

16.4.3 Helpfulness and Overrefusal Evaluations

To evaluate the tradeoff between improved safety and the overall helpfulness or overrefusal behavior of our models, we evaluate on 300 benign user requests from Alpaca [Taori et al., 2023]. To judge the quality of the model responses, we use GPT-4o-mini to judge examples as: (1) Compliant and Helpful, (2) Compliant and Unhelpful, (3) Partial Refusal, (4) Full refusal, and (5) Shortcircuit. Categories (1) and (2) represent model compliance, while categories (3)-(5) denote overrefusal in this setting. Therefore, desirable behavior is to minimize the number of overrefusals of categories (3)-(5) and to have more responses in category (1). The full judge details are deferred to Appendix 16.6.2.

Results are presented in Figure 16.3b. Our results show that Safety Pretraining does not come at a cost of helpfulness or overrefusal, while SafeBeam observes only a slight increase in overrefusal rate.

16.5 Discussion

The central claim that this work builds on is that *alignment is not unlearning*. Post-hoc methods, though widely adopted, fail to meaningfully remove harmful capabilities from language models—they merely redirect or suppress them. This brittleness becomes evident under benign finetuning, where aligned models can regress into unsafe behaviors with slight perturbations. These failures motivate a rethinking of safety: not as a post-hoc patch, but as a principle embedded from the start.

To design models that are natively safe, we take inspiration from the human learning process. Children are not exposed to the full spectrum of knowledge immediately. Instead, learning begins in tightly controlled, supervised environments—homes, classrooms—where safety and context are paramount. Complex and potentially harmful topics like violence, injustice, or discrimination are introduced only within structured settings that emphasize ethical reasoning and responsible interpretation.

Our empirical results support this view. Merely censoring data by removing all unsafe examples proved ineffective; models trained only on safe content were still brittle, unable to handle harmful prompts responsibly. Instead, the most robust safety gains came from interventions that mirrored real human pedagogical practices. These steps not only reduced attack success rates but also enhanced the model's ability to *understand* why a request is unsafe, not just that it is. In the long term, we envision safety pretraining as the foundation of responsible AI development. Just as human morality is built not through censorship but through careful instruction and internalization, safe language models must develop a native sense of harm and refusal.

16.6 Ablations

In earlier sections, we introduced a suite of data-centric and training-time interventions as part of our safety pretraining framework. To understand the contribution and importance of each intervention, we conduct a step-wise ablation to measure the impact of different data-centric interventions on safety, using Attack Success Rate (ASR) on GSM post benign finetuning as the metric (Figure 16.2).

16.6.1 Importance of Various Data-Centric Interventions

Training on only a safe subset hurts. A perhaps surprising result: filtering for safety alone increases risk. Switching from raw data to just score 0 examples leads to a rise in

ASR—from 38.8% to 43.8%. The model, having never seen unsafe patterns, fails to learn how to respond to them. This shows that removing risk isn't the same as building robustness.

Adding rephrased unsafe data helps. Instead of relying solely on removal, introducing rephrased unsafe content proves to be a far more effective alternative. By rewriting score 1–3 data in a context-aware, educational style, the model is exposed to sensitive topics framed with care. This leads to a notable drop in ASR: from 38.8% (raw data) to 33.6%. The rephrased examples help the model learn how to handle challenging topics not by avoiding them, but by engaging with them responsibly.

Refusal data adds strong guardrails. While rephrasing teaches responsible framing, it cannot cover the most toxic content (score 4 and score 5). For these cases, we introduce synthetic refusal completions that explicitly reject unsafe requests. These examples serve a complementary role: they define hard boundaries where dialogue must stop. Their impact is substantial—adding them reduces ASR further to 25.1%, an 8.5-point improvement.

Moral education helps understand why certain topics are sensitive. Beyond just knowing when to disengage, can a model learn why certain content is harmful in the first place? Our final intervention answers that question. By introducing moral education data—structured narratives and explanations about the risks and ethics behind unsafe behaviors—we move from rule-following to reasoning. This addition leads to further improvement with an ASR of 23.0%, and suggests that the model not only avoids harmful outputs but begins to internalize principles that guide safer generation.

SafeBeam using Harmfulness-Tags yields the strongest models. Steering generation away from harmful content at inference time using our SafeBeam decoding yields the strongest results, achieving the lowest ASR of 8.3%. In Figure 16.6.2 (Appendix 16.6.2), we provide an ablation where SafeBeam is applied to a standard pretrained model with harmfulness-tag injection, but without any rephrasing or contextualization. While this setup improves safety (ASR~10%), it significantly degrades helpfulness. This is because effective steering requires the presence of safe completions among the top beam candidates—something that is largely enabled by the earlier data-centric interventions such as rephrasing and recontextualization. Finally, we ablate the amount of `<potentially_unsafe_content>` used during pretraining in Appendix **??**, and find that injecting it into 5% of the tokens yields the best tradeoff between safety and utility.

16.6.2 <potentially_unsafe_content> Annotation During Pretraining vs. Finetuning

The primary goal of <potentially_unsafe_content> annotated pretraining is to induce a separation between the representations of safe and unsafe content by explicitly flagging unsafe content during training and priming the model accordingly whenever the input requires careful considerations. A natural question arises: can similar separation be achieved by only applying <potentially_unsafe_content> annotation during instruction fine-tuning (IFT), as explored recently in Xhonneux et al. [2025], Jain et al. [2024]. We compare our approach of pretraining with the <potentially_unsafe_content> (Sec. **??**) against standard pretraining followed by <potentially_unsafe_content> annotation during IFT only (Sec. **??**).

We observe a stark contrast in performance. <potentially_unsafe_content> annotated pretraining gives a significantly lower ASR of ~ 8% post benign finetuning on GSM8k. In contrast, surface level alignment with <potentially_unsafe_content> annotated IFT is quite brittle, with an almost 4x higher ASR of ~ 30%. Our results again highlight that it is critical to natively incorporate <potentially_unsafe_content> annotated pretraining, as surface level alignments using such approaches by finetuning are again brittle and not robust to even a small amount of benign finetuning. These findings validate the key premise of our work: post-hoc alignment methods do not amount to unlearning. Once such behaviors are internalized, they persist beyond surface-level alignment and reappear after benign instruction tuning.

16.7 Future Work and Conclusion

The work presented in this proposal addresses crucial aspects of training foundation models responsibly and efficiently. Looking ahead, there are several promising directions to build on this foundation:

- **Safety-Aware Data Filtering:** An immediate extension is to integrate *content moderation* into the data filtering pipeline. Chapters 4 and 5 dealt mainly with utility-based filtering (quality, redundancy). In future work, we plan to incorporate *toxicity and bias filters* so that models not only avoid wasting compute on low-quality data, but also avoid learning from harmful content. This could involve training classifiers or using lexicon-based methods to filter out hate speech, extreme adult content, or highly biased text/images from the pretraining corpora. The challenge is doing so while preserving the richness of data needed for broad coverage. Our scaling laws approach can be extended here: for example, how does filtering out toxic content affect model performance on downstream tasks, and where is the trade-off? By quantifying this, we can aim for datasets that are both high-quality and safe, without severely sacrificing capability.

- **Privacy-Preserving Pretraining:** The ultimate goal of the memorization and unlearning research (Chapters 2 and 3) is to enable training regimes where *privacy is preserved by design*. One future direction is to explore differentially private training at web scale—techniques that add noise or regularize training such that no single datum has undue influence. Traditional DP-SGD does not easily scale to the sizes of GPT-3 or CLIP, but insights from our memorization localization (e.g., focusing regularization on neurons that store memorized info) might yield more tractable methods. Additionally, improving unlearning algorithms (perhaps guided by our TOFU benchmark) is an ongoing effort. We aim to devise **efficient unlearning** methods that can adjust a model without a full retrain, proven effective by our rigorous metrics. This could involve meta-learning approaches where the model retains an intermediate representation of each training example (enabling removal) or using our example-tied dropout idea to simply silence memorization neurons for forgotten data. By the thesis defense, we hope to demonstrate a system where one can *quickly scrub* a foundation model of specific data and verify the scrubbing.

- **Foundation Model Data Transparency:** Inspired by our dataset attribution work, another future direction is developing **transparency tools** like *data sheets for models*. This includes automating the process of documenting what went into a model's

training data—listing major domains, perhaps even influential individual sources—and what the model is prone to memorize. Such transparency reports could be generated using a combination of dataset inference, memorization metrics, and provenance analysis. They would greatly aid users and auditors in understanding a model's potential biases or liabilities. We foresee techniques like ACR (Adversarial Compression Ratio) being used not just in research but as part of routine model audits: e.g., running a battery of known sensitive strings through a model to ensure none are memorized beyond a safe threshold [Schwarzschild et al., 2024b].

- **Scaling and Generalization of Synthetic Data Augmentation:** The WRAP approach in Chapter 6 demonstrated benefits for language models. Future work will explore *synthetic data generation for other modalities* (e.g., generating image variations or audio transcripts to aid multimodal model training). Studying the long-term dynamics of model-in-the-loop data generation is an exciting research avenue, especially considering how data diversity and quality interplay when model training breaches the trillion token mark.

In **conclusion**, this thesis aims to make foundation model training **more principled, transparent, and efficient**. By understanding what models remember and why, we can curb unwanted memorization and empower mechanisms to forget. By understanding what data truly matters for learning, we can curate and even create better corpora that yield equal or greater performance with less computational effort. These advances are increasingly important as models grow ever larger and more integrated into society. We want the next generation of foundation models to be *responsible citizens* of the AI ecosystem—trained on data that respects privacy and ownership, unlikely to regurgitate secrets or toxic content—and to be *efficient students*, extracting maximal knowledge from each byte of data and FLOP of compute expended.

The contributions outlined in this proposal make significant strides toward those goals. Going forward, we will continue to interweave ideas from privacy, learning theory, and data engineering to push the boundaries of what's possible in foundation model training. By the end of this thesis, we hope to deliver both theoretically grounded insights and practical tools that help align the development of large-scale AI with societal values and resource realities. In a world where "more data, bigger model" has been the default mantra, we offer a nuanced counterpoint: **better data and smarter training** can achieve more with less, and can do so while upholding the principles of responsibility that our increasingly AI-driven world demands.

Chapter A

LLM Dataset Inference

Paper: LLM Dataset Inference
Reference: [Maini et al., 2024b]
Link:

Broader Impact

The use of large amounts of text scrapes from the web to train large language models (LLMs) has recently sparked significant privacy concerns. Including personally identifiable information (PII) and copyrighted material in these datasets has resulted in legal disputes, such as the lawsuit between The New York Times and OpenAI, among others. These disputes emphasize the problem of using copyrighted content without proper attribution, potentially infringing on creators' rights and discouraging future artistic work due to the lack of financial compensation for publicly available content. This scenario underscores the technical challenge of identifying data used to train machine learning models. Despite the legal uncertainties, this task is essential for understanding LLM behavior and ensuring data privacy. In our work, we propose a new *dataset inference* method to accurately identify the datasets used to train large language models. Our LLM dataset inference method is statistically grounded and we thoroughly evaluate our approach.

Compute

To aggregate various membership inference attacks, we required performing forward passes for both suspect and reference models. We need enough GPU memory to load a model. To allow for large batch forward passes, we utilized NVIDIA A6000 48GB machines for aggregating such metrics. We used a total of 4 machines at any given point to speed up the aggregation of metrics.

Additional Experiments

Setup. Note that for the perturbation-based MIAs (*e.g.,* [Mattern et al., 2023]), we use the implementation of the perturbations from the NLAugmenter library [Dhole et al., 2023].

Performance of MIAs. We provide an extended version of the performance of various MIAs on different subsets of the Pile dataset in Fig. A.1. For an MIA to be effective, it must achieve an AUC significantly higher than 0.5. Only a few methods meet this standard for certain datasets, and their success is not consistent across different datasets. We note that, for example, the DetectGPT [Bao et al., 2024] performs significantly better on a few datasets, for example, bookcorpus2 and books3, than any other metric.

Feature Selection. We provide an extended version of the importance of features used in dataset inference in Fig. A.2.

Additional ablation studies on model size, processing technique, and number of data points. We analyze the number of required data points for successful dataset inference, with processing techniques used in dataset inference and different sizes of LLMs, in Fig. A.3

and Fig. A.4 respectively. To visualize the distribution of p-values impacted by the number of data points and size of the models, violin plots are presented in Fig. A.5.

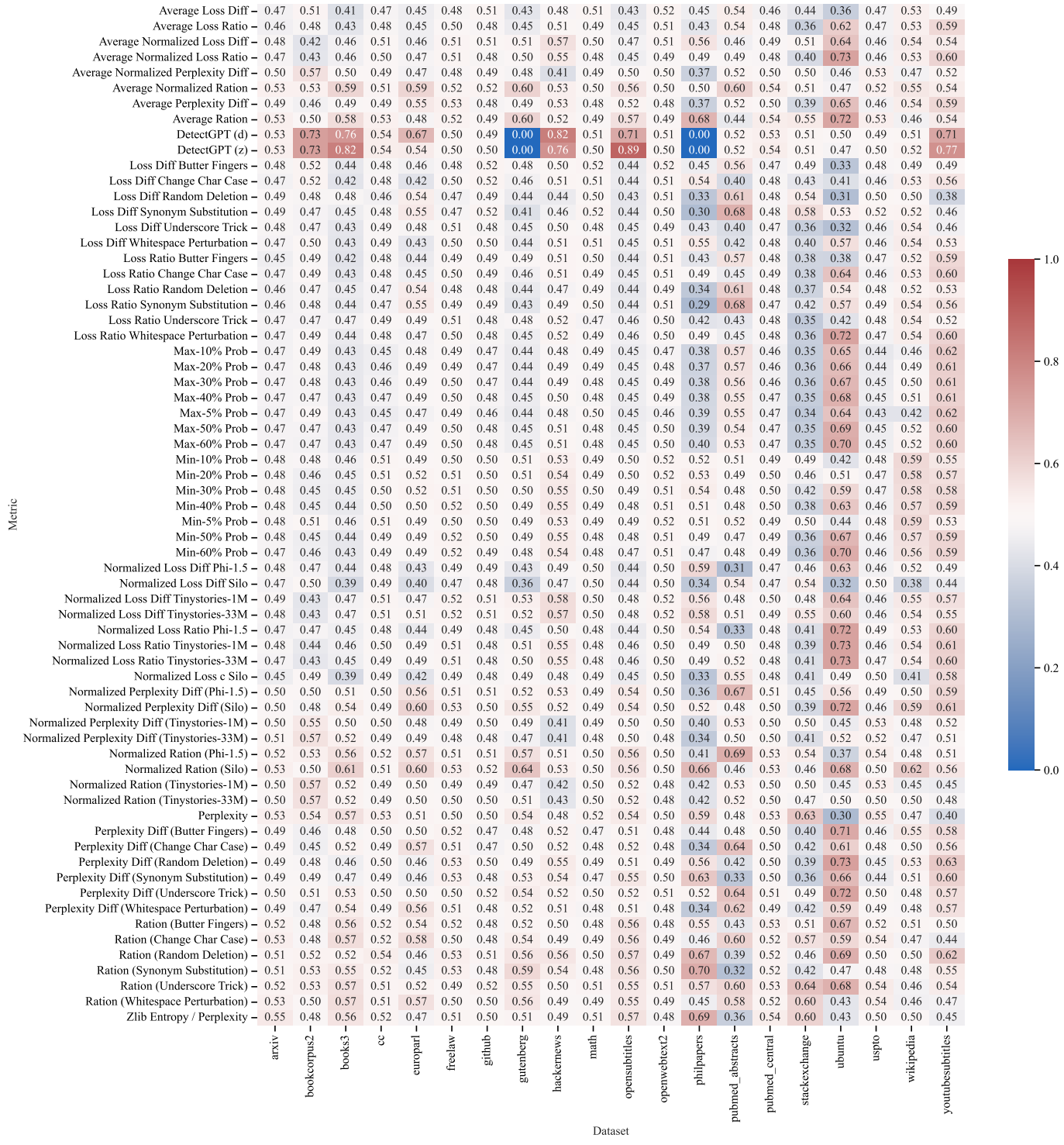


Figure A.1: Extended version of Figure 2.3 with more types of membership inference attacks. The results are consistent: none of the membership inference attacks can consistently achieve high ROAUC across different datasets.

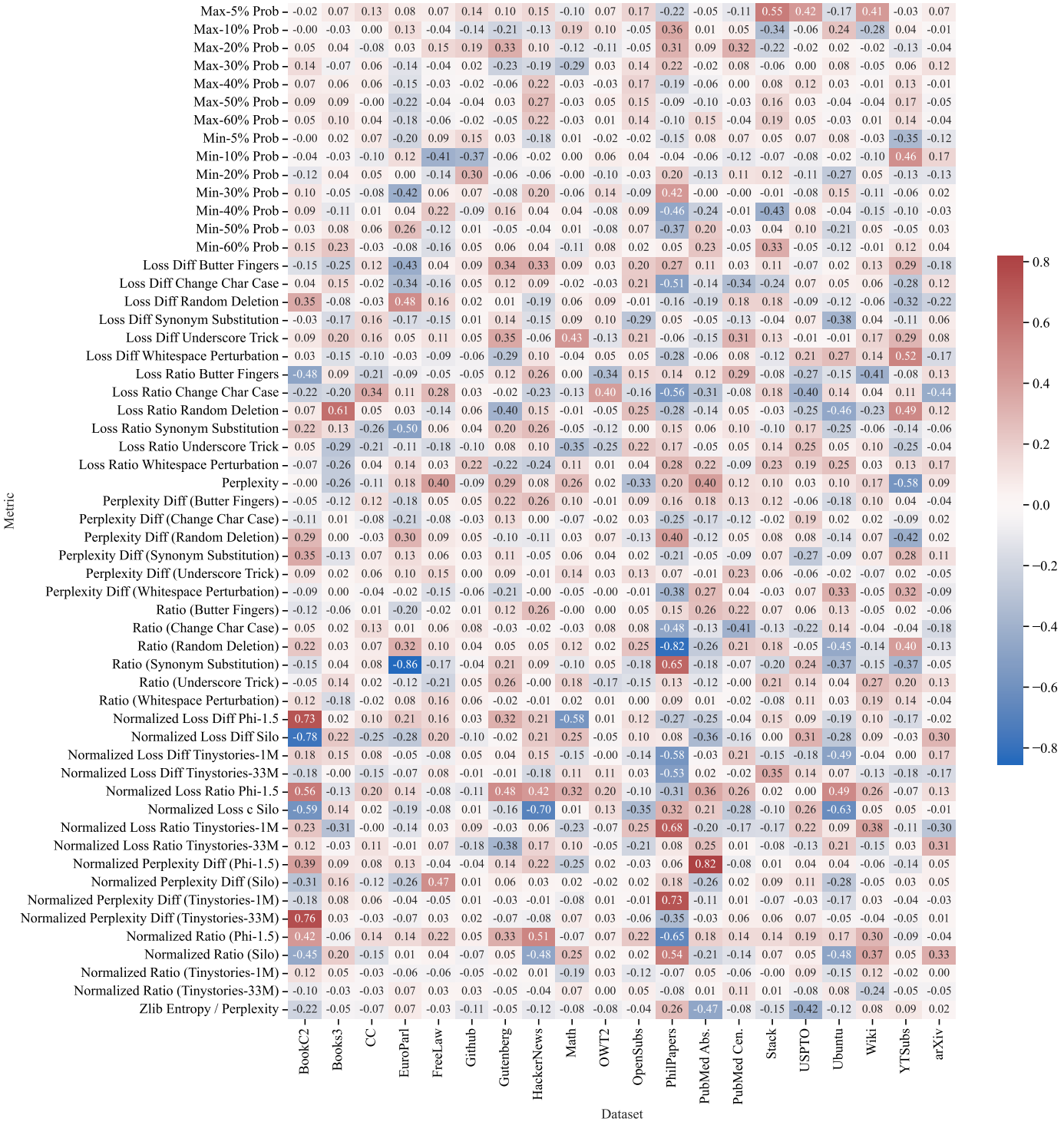
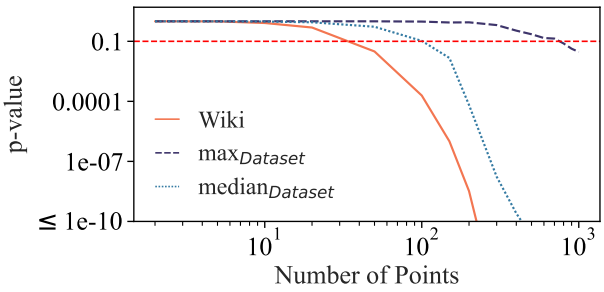
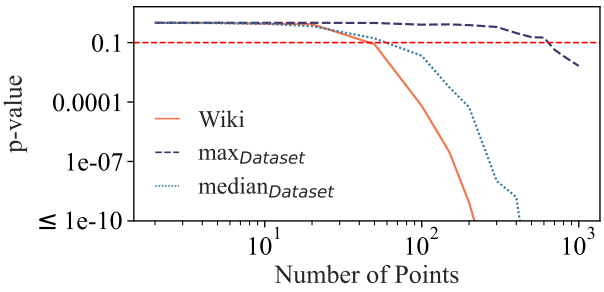


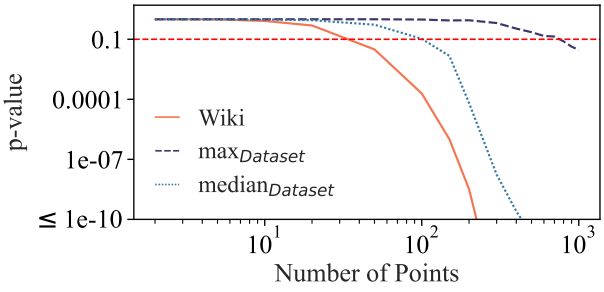
Figure A.2: Extended version of Figure 2.5a with more features that we leverage for the LLM dataset inference. The results are consistent—one can see most features contribute positively to dataset inference for some datasets, while negatively to other datasets.



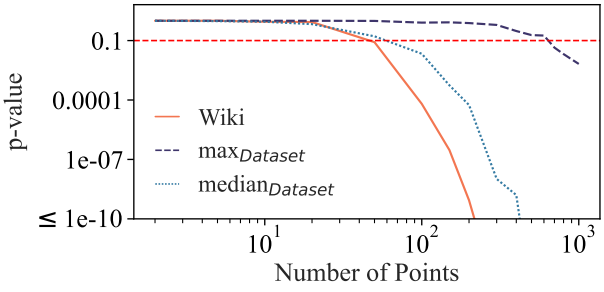
(a) Mean Correction



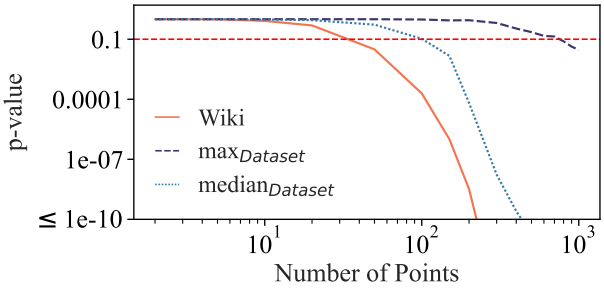
(b) Mean Correction (Normalized)



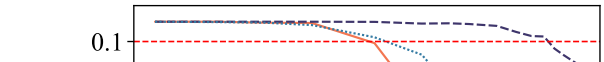
(c) Outlier Removal

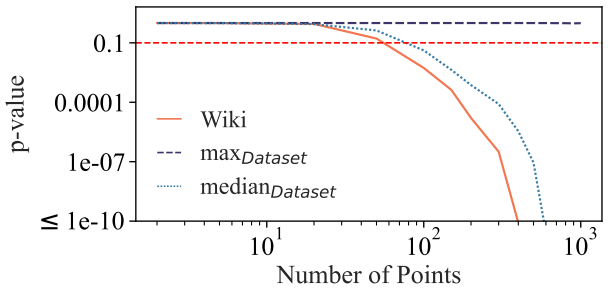


(d) Outlier Removal (Normalized)

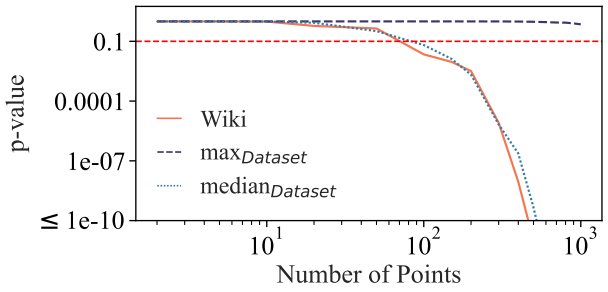


(e) Both

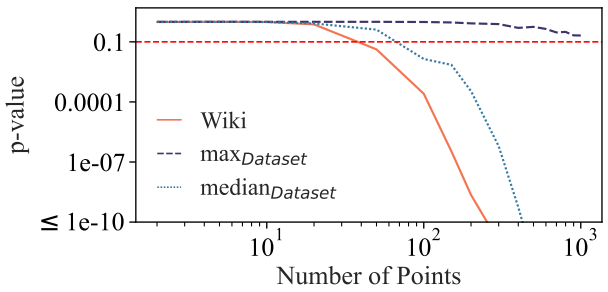




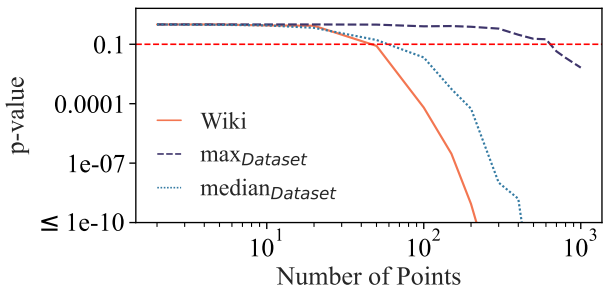
(a) Pythia-410m



(b) Pythia-1.4b

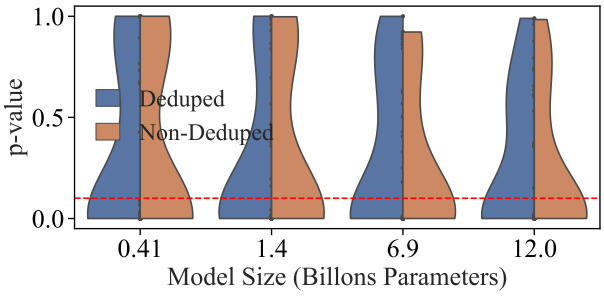


(c) Pythia-6.9b

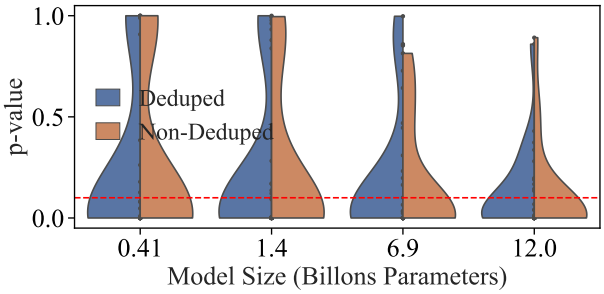


(d) Pythia-12b

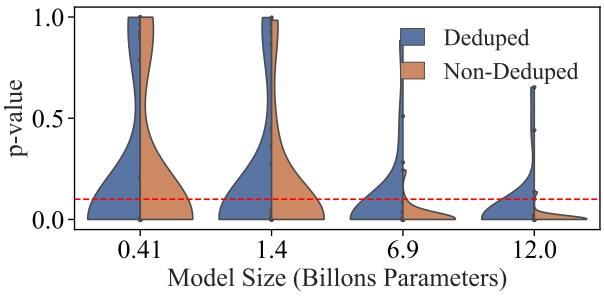
Figure A.4: Extended version of Figure 2.6a with different model sizes. Similarly, the three curves correspond to the median, and maximal p-value across the datasets in Pile, and the p-value of Wikipedia dataset. An interesting observation is that the curves for median p-values and Wikipedia's p-value stay similar with respect to models of different sizes. However, this is not the case for the max p-value curve. This indicates that dataset inference does not rely on a large number of data points or model parameters for most datasets, whereas they may be necessary for some particular datasets.



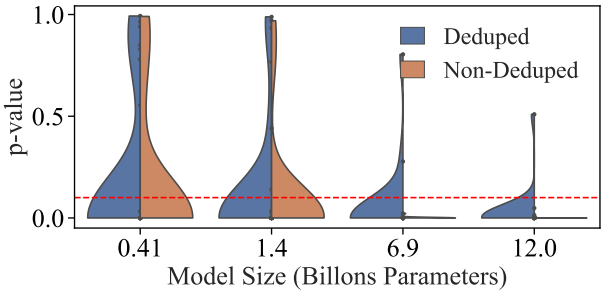
(a) 100 data points



(b) 200 data points



(c) 500 data points



(d) 1000 data points

Figure A.5: Extended version of Figure 2.6b with different numbers of data points used in dataset inference. As expected, a larger number of data points allow the p-values to be more concentrated below the threshold, especially when the model size is large.

Chapter B

STAMP

Paper: STAMP: Spotting Training Artifacts through Watermarked Pairs
Reference:
Link:

Additional Results

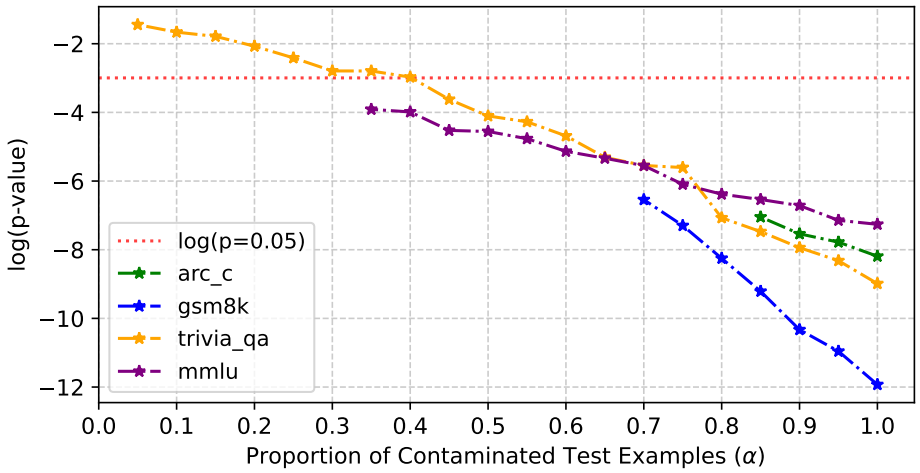


Figure B.1: Log p-value vs proportion of benchmark that is contaminated. We plot the log p-value against the proportion of test examples that are leaked to analyze the sensitivity of our test to detect contaminated in scenarios where the benchmark is only partially contaminated (lower is better).

Detecting Partial Contamination In practice, benchmarks may be partially contaminated, where only a subset of test examples appears in the pretraining corpora. Understanding the impact of partial contamination is critical because benchmark owners cannot identify which specific test examples have been leaked. This study complements our earlier analysis in Section 3.5.4, by focusing on the sensitivity of our approach under varying proportions (α) of contaminated examples within a fixed benchmark size (n).

Our results in Figure B.1 highlight that as α increases the detection strength improves, with p-values dropping below 10^{-3} when majority of the benchmark is contaminated. We also observe that STAMPreliably detects contaminated even when only 40% of the

test examples are contaminated. Our findings confirm that STAMPsuccessfully identifies contamination with high statistical significance, even in scenarios of partial contamination.

P-SP Metric

To validate semantic preservation in our watermarking process, we employ P-SP [Wieting et al., 2022], a state-of-the-art semantic similarity model. P-SP uses embedding averaging trained on a large corpus of filtered paraphrase data, and has been shown to effectively distinguish between true paraphrases and unrelated text. As evidenced by Krishna et al. [2023], P-SP assigns an average score of 0.76 to human-created paraphrases in the PAR3 dataset [Thai et al., 2022], while random paragraph pairs from the same book score only 0.09. Table 3.6 reports the average P-SP scores between original benchmarks and their watermarked versions across 9 random *hash keys*. Our watermarked versions achieve high P-SP scores (0.83-0.95) across all benchmarks, substantially exceeding the average score for human paraphrases, indicating strong semantic preservation.

Bag-of-Words Classifier

We train a random forest classifier on the *bag-of-words* feature representations for the datasets. The classifier is trained on 80% of the member and non-member sets, with evaluation performed on the remaining 20%. Results are aggregated over a 5-fold cross-validation. The detailed results are presented in Table 3.3.

Perplexity

Perplexity (*PPL*) measures how well a language model predicts a given text sequence S, with lower values indicating better prediction. For an auto-regressive language model θ and text sequence S, tokenized as a sequence of N tokens $\{s_1, \ldots, s_N\}$, perplexity is computed as the exponent of the loss. Formally:

$$PPL_\theta(S) = \exp\left(\mathcal{L}_\theta(S)\right) \tag{B.1}$$

Where the loss $\mathcal{L}_\theta$ is defined as:

$$\mathcal{L}_\theta(S) = -\frac{1}{N}\sum_{i=1}^{N} \log\left(\mathcal{P}_\theta(s_i|s_{<i})\right) \tag{B.2}$$

Here $\mathcal{P}_\theta(s_i|s_{<i})$ denotes the predicted probability for token s_i by the language model θ given the context of previous tokens $\{s_1, \ldots, s_{i-1}\}$.

Pretraining Details

We continually pretrain Pythia 1B on intentionally contaminated OpenWebText Test case instances from the benchmark were randomly inserted between documents from OpenWebText. We trained for 1 epoch of 46000 steps with an effective batch size of 144 sequences and sequence length of 1024 tokens. We used the AdamW optimizer [Loshchilov and Hutter, 2019] with a learning rate of 10^{-4}, $(\beta_1, \beta_2) = (0.9, 0.999)$ and no weight decay.

Watermark for Large Language Models

In work, we use the prominent KGW [Kirchenbauer et al., 2023] watermarking scheme. KGW scheme uses a hash function that takes the context (preceding tokens) and a hash key h to partition the vocabulary V into two disjoint sets at each generation step: a green list G and a red list R. Formally, for a language model $\mathcal{M}$ with vocabulary V, and a prefix comprising tokens $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n$, the scheme involves first computing the logits $\mathcal{M}(\mathbf{w}_1 \ldots, \mathbf{w}_n) = (l_1, \ldots, l_{|V|})$ of the language model that would ordinarily be used to predict the subsequent token. As per a hyper-parameter k, the last k tokens, $\mathbf{w}_{n-k+1}$ to $\mathbf{w}_n$, are then fed to a pseudo-random function F to partition V into a green list G and a red list R such that $|G| + |R| = |V|$. Finally, the logits corresponding to the tokens in the green list, G, are boosted by δ ($\delta > 0$). Specifically, in our work we set $k = 1$ and $\delta = 1.0$ as the chose hyperparameters. The watermark can then be detected through a one-proportion z-test on the fraction of green tokens in the generated text.

Related Works

Our works relates to a large literature of work on membership inference (Sec. B), dataset membership (Sec. B) and test-set contamination detection (Sec. B) in large language models.

Watermarking for Dataset Membership Liu et al. [2023c] propose TextMarker, a backdoor-based membership inference technique for protection of *classification datasets*. Their approach watermarks each original sample (x, y) by inserting specific triggers (character or word-level substitutions) into x, creating a backdoored sample x_t. They then assign an altered target label y_t ($y_t \neq y$) to this modified input. Membership detection is performed by testing whether a model f produces the watermarked label with high probability: $\Pr(f(x_t) = y_t)$. While effective, TextMarker is specifically designed for classification datasets and it is not trivial to extend it to other kinds of benchmarks or the broader problem of dataset membership detection, limiting its real-world applicability.

Recent work has proposed detecting watermark signals in suspect model outputs to

determine dataset membership. Waterfall [Lau et al., 2024] enables creators to watermark their text using a robust watermarking scheme that leverages an LLM for rephrasing. They further demonstrate that watermarks in fine-tuning data, persist in downstream LLM outputs, allowing for membership detection. While both approaches use LLM-based rephrasing, Waterfall relies on detecting watermarks directly in model generations, whereas our method uses multiple watermarked versions to detect perplexity divergences. While Waterfall is effective for text watermarking, it has limited utility for pretraining membership detection. It relies on strong overfitting and presumes that the model is trained on datasets over multiple epochs. Additionally, their approach is not applicable to short text segments commonly found in benchmarks. Another recent contemporaneous study [Sander et al., 2025] proposes a similar approach where watermarks are embedded in benchmarks by reformulating the original questions with a watermarked LLM. While employing a similar setup, their detection approach differs substantially from ours. Their method relies on detecting model overfitting on the green tokens in the watermarked benchmark to prove contamination, whereas our approach compares perplexity differences between the publicly released benchmarks and private versions watermarked with different keys.

Membership Inference Membership inference, initially proposed by Shokri et al. [2016], is a long-standing problem in machine learning: given a data point and a machine learning model, determine whether that data point was used to train the model. MIAs for LLMs are broadly based on applying pre-defined thresholds to membership scores that are typically based on *loss-based* metrics. We briefly describe the specific membership scores proposed by different MIAs that we employ in our experiments.

- Perplexity: Proposed by Yeom et al. [2018b], this MIA uses loss (perplexity in the context of LMs) as the scoring metric. However, this approach suffers from high false positives as it tends to classify naturally predictable sequences as members of the training set.
- zlib Entropy [Carlini et al., 2021c] computes a score by taking the ratio between the model's perplexity and the zlib compression size of the text. Lower ratios indicate potential membership in the training data.
- Min-K% [Shi et al., 2024b] computes the score by averaging the probabilities of the k% least likely tokens in a sequence. By focusing on the least likely tokens, it aims to solve the false positive problem with perplexity.
- Min-K%++ [Zhang et al., 2024b] compares the probability of the target token with the

expected probability of all tokens within the vocabulary. It is based on the insight that each training token will tend to have higher probability relative to many other candidate tokens in the vocabulary.

- DC-PPD [Zhang et al., 2024e] computes the divergence between the token probability distribution and the token frequency distribution for detection.

Detecting Dataset Membership Detecting dataset membership addresses the challenge of detecting whether a given dataset was used by LLM developers in pretraining. Unlike membership inference attacks (MIAs), which focus on identifying whether individual sequences were included in a model's training data, dataset membership concerns verifying the inclusion of a collection of documents.

Wei et al. [2024] propose a hypothesis-testing approach to detect membership by inserting random sequences or Unicode character substitutions as data watermarks. This method works by testing the model's preference for the inserted data watermarks against other random data watermarks. First, their proposed watermarks can impact machine readability, affecting search engine indexing and retrieval-augmented generation (RAG) pipelines. More critically, unicode substitutions can significantly alter tokenization processes, potentially compromising the utility of evaluation benchmarks. Although these limitations may be manageable for some creators, Our approach offers an alternative that better preserves content quality while maintaining detection capability. Another recent proposal [Maini et al., 2024c] is to selectively combining MIAs that provide positive signal for a given distribution, and aggregating them to perform a statistical test on a given dataset. Their method assumes access to a *validation* set drawn from the same distribution as the target dataset and unseen by the model–a requirement that can be challenging to satisfy in many practical scenarios.

Meeus et al. [2024a] propose inserting "copyright traps" into documents to enhance document-level membership inference for smaller models that lack natural memorization. Liu et al. [2023c] introduce a backdoor-based dataset inference approach. However, these methods rely on heuristics and do not provide the false positive guarantees that hypothesis-testing-based approaches offer.

Recent studies [Maini et al., 2024c, Duan et al., 2024b, Das et al., 2024, Meeus et al., 2024b] suggest that detecting sequence level membership in LLMs trained on trillions of tokens in a single epoch is likely infeasible. These studies also highlight the limited efficacy of MIAs for LLMs, showing that such approaches barely outperform random guessing. Moreover, the apparent success of MIAs in certain scenarios can often be attributed to distributional differences between the *member* and *non-member* sets used in evaluations, rather than their ability to reliably infer true membership.

Test Set Contamination Detection There have been a few recent third-party approaches that are focused on detecting test-set contamination in LLMs. Heuristic prompting-based methods [Sainz et al., 2023b, Golchin and Surdeanu, 2024] attempt to detect contamination by prompting models to reproduce exact or near-exact test examples. Reproducing verbatim examples requires a high level of memorization which typically requires a high duplication of test examples [Carlini et al., 2021c] and strong memorization capabilities typically absent in smaller models [Meeus et al., 2024a]. The heuristic nature of these approaches prevents them from providing a statistical evidence of contamination.

Statistical approaches to detect contamination are limited. Oren et al. [2024] build on the principle that in absence of data contamination, all orderings of an *exchangeable* test set should be equally likely. Their work relies on the strong assumption of metadata contamination (canonical ordering of the dataset)–a presumption that can often be violated. Another recent proposal [Zhang et al., 2024a] uses a statistical test to compare model confidence on original test instances and their rephrased counterparts. However, as discussed earlier, their null hypothesis can be invalid due to LLMs' inherent bias towards machine-generated content.

Radioactivity of Watermarks

Sander et al. [2024] proposed methods to detect when watermarked texts are used as fine-tuning data for an LLM. Their approach is based on the insight that training on watermarked texts leaves detectable traces of the watermark signal in the resulting model due to token-level overfitting. In a recent contemporaneous study, Sander et al. [2025] extended this approach to detect benchmark contamination. Specifically, they propose watermarking benchmarks before release and later detecting traces left by the watermarked benchmarks through a statistical test. Since the statistical test relies on token-level overfitting, their approach requires duplication and stronger watermarks, which introduce more distortion into the rephrasings. Additionally, the tokenizer-dependent nature of detecting watermarks limits the applicability of their approach, as the rephrasing model and contaminated model need to share the same tokenizer.

Given the requirement that the rephrasing and contaminated LLM should share the same tokenizer, we conduct additional controlled experiments comparing the approaches. We rephrase with Llama-3.1-8B Instruct [AI@Meta, 2024] with top-p sampling with $p = 0.7$ and temperature = 0.5, matching the sampling parameters used in the original study. For watermarking, we use KGW [Kirchenbauer et al., 2024] scheme, with context window of size 2, split ratio (γ) of 0.5 & and boosting value (δ) of 2. We create a contaminated corpus of 2 billion tokens following our methodology in Section 3.5.2, but with a duplication count

of 4, meaning each benchmark sample is inserted four times in the pretraining corpora.

We compare STAMPwith Sander et al. [2025] for detecting benchmark contamination in Table **??**. With a moderate watermarking strength ($\delta = 2.0$) and repetition count of 4, the radioactivity based approach fails to detect contamination of watermarked test examples, while STAMPachieves significantly low p-values. These results align with the original paper's findings, which indicated that their method requires around 16 repetitions to achieve low p-values.

Case Study: Detecting Research Paper Abstracts in Pretraining Data

To demonstrate the broader applicability of STAMPfor detecting dataset membership across different forms of content, we explore its effectiveness in detecting membership of abstracts of papers from EMNLP '24 proceedings [emn, 2001]. We evaluate both the preservation of academic writing quality in watermarked abstracts and the effectiveness of STAMPin detecting their inclusion in training data.

Experimental Setup. We sample 500 papers from EMNLP 2024 proceedings and create watermarked versions of their abstracts following our methodology from Section 3.4. The prompt template and examples of rephrased abstracts are presented in Appendix B and Appendix B respectively. To evaluate detection capability, we perform controlled experiments on the Pythia 1B model [Biderman et al., 2023b] through continual pretraining. The pretraining corpora consists of a mixture of the public watermarked versions of these abstracts and a subset of OpenWebText (approximately 3 billion tokens). The abstracts comprise approximately 100K tokens, representing just 0.003% of the pretraining corpus.

Results. Table **??** demonstrates STAMP's effectiveness in detecting dataset membership. Our approach achieves a near-zero p-value ($\approx 10^{-12}$), indicating strong statistical evidence of membership. For comparison, LLM DI [Maini et al., 2024c] achieves a p-value of 0.05 with access to a *validation* set of unseen abstracts from the same conference and is unable to detect membership using the privately held counterparts of the same abstracts included in the pretraining data as the *validation* set. In Table 3 we evaluate state-of-the-art MIAs and finding that they perform no better than random chance (AUROC ≈ 0.5). Our findings corroborate with recent studies [Duan et al., 2024b, Maini et al., 2024c, Das et al., 2024] that highlight the failure of sequence level MIAs on LLMs.

Quality Evaluation. To evaluate the quality of watermarked abstracts, we use GPT-4 [OpenAI, 2023] as a judge following the prompt template in Figure B. Each abstract was

classified into one of five quality tiers. Our analysis shows that 82.7% of the watermarked abstracts were rated as *preferred* and 16.3% as *acceptable* indicating that 99% maintain high academic quality. Only 1% required *minor revisions*, with none requiring *major revisions* or deemed *inadequate*.

Since LLMs often exhibit systematic preferences for LLM-generated text over human-written content [Liu et al., 2023d, Mishra et al., 2023, Laurito et al., 2024], we additionally conduct a human study involving the original authors. We asked 40 authors to rate watermarked versions of their own abstracts using the same quality tiers. The human evaluation strongly corroborates our automatic assessment, with most watermarked versions being *preferred* or *acceptable*: 4 authors *preferred* the watermarked version, 24 authors rated the watermarked abstracts as *acceptable*, 11 indicated the text required *minor revisions* and just 1 indicating that their rephrased abstract requires major edits.

Additionally, we measure semantic preservation using the P-SP metric [Wieting et al., 2022], finding an average score of **0.95** between original and watermarked abstracts, demonstrating strong semantic similarity.

Prompt Template to Evaluate Quality of the Rephrased Abstracts using GPT4

```
You will be given an original abstract and its rephrased version.
Your task is to evaluate the quality of abstract rewrites for ML
research paper based on:

1. Meaning Preservation
2. Clarity
3. Technical Accuracy

Evaluate the rewritten abstract and assign one of these ratings:
- Preferred: The rewrite improves upon the original in terms of
clarity and readability while maintaining full technical accuracy.
- Acceptable: The rewrite matches the original in quality and could
serve as a direct replacement without requiring changes.
- Minor Revisions: The rewrite is promising but requires minor edits
to reach the original's quality.
- Major Revisions: The rewrite has significant issues with meaning
preservation, clarity, or technical accuracy and requires major
edits.
- Inadequate: The rewrite fails to convey the original research
effectively due to critical flaws in meaning, clarity, or technical
accuracy.

Here are the abstracts:

Original Abstract: {original_abstract}
Rephrased Abstract: {watermarked_abstract}

Provide a short explanation of your rating, followed by your final
rating in the format:
Final Rating: {rating}
```

Case Study: Detecting ML Blog Posts in Pretraining Data

The inclusion of copyrighted material in LLM training data has emerged as a significant concern, leading to legal disputes, such as the lawsuit between New York Times and

OpenAI [NewYorkTimes, 2023], among others. Through a case study, we demonstrate how STAMPcan help creators detect potential unauthorized use of their content in model training. Specifically, we use STAMPto detect the membership of the popular AI Snake Oil newsletter [Narayanan and Kapoor, 2023].

Experimental Setup. We collect 56 blogs from the newsletter, creating watermarked versions of each newsletter using, the prompt template is presented in Figure B. We randomly select a subset of 44 blogs that we include in pretraining corpora and keep the remaining 12 blogs as a *validation* set that is unseen by the model. To evaluate detection capability, we perform controlled experiments on the Pythia 1B model [Biderman et al., 2023b] through continual pretraining. The pretraining corpora consists of a mixture of the public watermarked versions of these abstracts and a subset of OpenWebText (approximately 3 billion tokens). The abstracts comprise approximately 94K tokens, representing just 0.003% of the pretraining corpus.

Method	P-value (↓)
LLM DI [Maini et al., 2024c] (1)	0.44
LLM DI [Maini et al., 2024c] (2)	0.58
STAMP(w/o paired tests)	0.07
STAMP	**2.4e-3**

Table B.1: Comparison of different approaches for detecting membership of AI Snake Oil. Bold indicates statistically significant results ($p < 0.05$). Our approach results in lower p-values compared to other approaches (lower is better).

Results. Table **??** demonstrates STAMP's effectiveness in detecting dataset membership for the blog articles. LLM DI is unable to detect membership under the two different choices of validation set: (1) with the private rephrases of the same 44 blog posts as the validation set, and (2) with the version of the *held out* set of 12 blog posts that is watermarking using the public key. In Table 3 we evaluate state-of-the-art MIAs and finding that they perform no better than random chance (AUROC ≈ 0.5). Our findings corroborate with recent studies [Duan et al., 2024b, Maini et al., 2024c, Das et al., 2024] that highlight the failure of sequence level MIAs on LLMs.

Prompt Templates for Rephrasing

In this section, we outline the prompts used with LLaMA-3 70B [AI@Meta, 2024] to generate watermarked versions of the documents used in our experiments.

Prompt Template for Rephrasing Benchmarks

```
Rephrase the question given below. Ensure you keep all details
present in the original, without omitting anything or adding any
extra information not present in the original question.

Question: What is the main energy source for deep ocean currents
that move large volumes of water around the planet?

Your response should end with "Rephrased Question: [rephrased
question]"
```

Prompt Template for Rephrasing Abstracts

```
Rephrase the abstract of a ML research paper given below following
these strict guidelines:

PRESERVE:
- All technical details and findings
- Original tone of the abstract

AVOID:
- Adding interpretive language not present in the original abstract
- Removing any details
- Changing meaning or emphasis

Abstract: {original_abstract}

Your response should end with "Rephrased Abstract:
{rephrased_abstract}"
```

Prompt Template for Rephrasing Blogs

```
Rephrase the below paragraph from an AI newsletter while maintaining
coherent flow between paragraphs.  Here are your instructions:

1.  I will provide the previous paragraph (marked as CONTEXT) and the
current paragraph to rephrase (marked as TARGET).
2.Your task is to:
- Rephrase the TARGET paragraph so it flows naturally from the
previous paragraph (CONTEXT)
- Keep the same tone and emphasis as the original paragraph
-Preserve the technical details present in the original paragraph
- Do not add any extra information not present in the original
paragraph
- Avoid making sentences wordier or adding interpretive language

3.  Format your response as:  REPHRASED PARAGRAPH: [your rephrased
version]

Context:  {context}
Paragraph:  {paragraph}
```

Watermarked Examples

Watermarked Test Sets

TriviaQA

Original Question: Which enduring cartoon character was created by Bob Clampett for the 1938 cartoon Porky's Hare Hunt?

Rephrased Question: Which long-lasting cartoon character was originally created by Bob Clampett for the 1938 cartoon titled 'Porky's Hare Hunt'?

Original Question: Which US state lends its name to a baked pudding, made with ice cream, sponge and meringue?

Rephrased Question: Which US state is the namesake of a baked pudding that consists of sponge, meringue, and ice cream?

ARC Challenge

Original Question: Company X makes 100 custom buses each year. Company Y makes 10,000 of one type of bus each year. Which of the following is the most likely reason a customer would buy a bus from company X instead of company Y?

Rephrased Question: What is the most probable reason a customer would choose to purchase a bus from Company X, which produces 100 custom buses annually, over Company Y, which manufactures 10,000 buses of a single type each year?

Original Question: Sugars are necessary for human cell function. Which of the following are human cells not capable of doing?

Rephrased Question: Given that sugars are necessary for human cell function, what is it that human cells are unable to do?

MMLU

Original Question: Noradrenaline is the neurotransmitter between which of the two structures below?

Rephrased Question: Between which two structures listed below does noradrenaline act as the neurotransmitter?

Original Question: On which surfaces of the teeth is dental plaque most likely to accumulate in the mouth of a patient with poor oral hygiene?

Rephrased Question: In a patient with poor oral hygiene, on which surfaces of the teeth is dental plaque accumulation most probable in the mouth?

GSM8K

Original Question: Darrell and Allen's ages are in the ratio of 7:11. If their total age now is 162, calculate Allen's age 10 years from now.

Rephrased Question: If the current ages of Darrell and Allen are in a 7:11 ratio and their combined age at present is 162, what will be Allen's age a decade from now.

Original Question: Indras has 6 letters in her name. Her sister's name has 4 more letters than half of the letters in Indras' name. How many letters are in Indras and her sister's names?

Rephrased Question: If Indras' name contains 6 letters, and her sister's name has a number of letters that is 4 more than half the number of letters in Indras' name, then how many letters are there in Indras' and her sister's names combined?

Watermarked Paper Abstracts

Tree of Problems: Improving structured problem solving with comp
[Zebaze et al., 2024]

Original Abstract:

Large Language Models (LLMs) have demonstrated remarkable pe
across multipletasks through in-context learning. For compl
reasoning tasks that require step-by-step thinking, Chain-o
(CoT) prompting has given impressive results, especially whe
with self-consistency. Nonetheless, some tasks remain parti
difficult for LLMs to solve. Tree of Thoughts (ToT) and Gra
of Thoughts (GoT) emerged as alternatives, dividing the comp
problem into paths of subproblems. In this paper, we propos
of Problems (ToP), a simpler version of ToT, which we hypoth
can work better for complex tasks that can be divided into i
subtasks. Our empirical results show that our approach outp
ToT and GoT, and in addition per forms better than CoT on co
reasoning tasks. All code for this paper will be made avail

Watermarked Abstract:

Large Language Models (LLMs) have achieved notable performan
multiple tasks using in-context learning. For complex reaso
tasks requiring step-by-step thinking, Chain-of-Thought (CoT
prompting, especially when combined with self-consistency, h
yielded impressive results. However, certain tasks continue
significant challenges for LLMs. Tree of Thoughts (ToT) and
of Thoughts (GoT) have emerged as alternatives by breaking d
complex problems into subproblem paths. This paper introduc
of Problems (ToP), a simplified ToT version that we hypothes
be more effective for complex tasks comprising identical sub
Our empirical findings indicate that ToP outperforms both To
and also surpasses CoT on complex reasoning tasks. The code
this paper will be made available.

"A good pun is its own reword": Can Large Language Models Understand Puns? [Xu et al., 2024]

Original Abstract:

Puns play a vital role in academic research due to their distinct structure and clear definition, which aid in the comprehensive analysis of linguistic humor. However, the understanding of puns in large language models (LLMs) has not been thoroughly examined, limiting their use in creative writing and humor creation. In this paper, we leverage three popular tasks, i.e., pun recognition, explanation and generation to systematically evaluate the capabilities of LLMs in pun understanding. In addition to adopting the automated evaluation metrics from prior research, we introduce new evaluation methods and metrics that are better suited to the in-context learning paradigm of LLMs. These new metrics offer a more rigorous assessment of an LLM's ability to understand puns and align more closely with human cognition than previous metrics. Our findings reveal the "lazy pun generation" pattern and identify the primary challenges LLMs encounter in understanding puns.

Watermarked Abstract:

Puns, due to their distinct structure and clear definition, play a vital role in academic research on linguistic humor analysis. Despite their importance, the understanding of puns in large language models (LLMs) remains understudied, which hinders their application in creative writing and humor generation. This paper employs three tasks - pun recognition, explanation, and generation - to systematically evaluate LLMs' capabilities in pun understanding. Building upon prior research's automated evaluation metrics, we propose new evaluation methods and metrics tailored to the in-context learning paradigm of LLMs. These novel metrics provide a more rigorous assessment of an LLM's pun understanding ability and better align with human cognition. Our results expose the "lazy pun generation" pattern and pinpoint the primary challenges LLMs face in understanding puns.

MiniCheck: Efficient Fact-Checking of LLMs on Grounding Documents [Tang et al., 2024]

Original Abstract:

Recognizing if LLM output can be grounded in evidence is central to many tasks in NLP: retrieval-augmented generation, summarization, document-grounded dialogue, and more. Current approaches to this kind of fact-checking are based on verifying each piece of a model generation against potential evidence using an LLM. However, this process can be very computationally expensive, requiring many calls to a model to check a single response. In this work, we show how to build small fact-checking models that have GPT-4-level performance but for 400x lower cost. We do this by constructing synthetic training data with GPT-4, which involves creating realistic yet challenging instances of factual errors via a structured generation procedure. Training on this data teaches models to check each fact in the claim and recognize synthesis of information across sentences. For evaluation, we unify datasets from recent work on fact-checking and grounding LLM generations into a new benchmark, LLM-AggreFact. Our best system MiniCheck-FT5 (770M parameters) outperforms all systems of comparable size and reaches GPT-4 accuracy. We release LLM-AggreFact, code for data synthesis, and models.

Watermarked Abstract:

Verifying the evidential grounding of LLM outputs is crucial for various NLP tasks, including retrieval-augmented generation, summarization, and document-grounded dialogue. Existing fact-checking approaches typically involve verifying each component of model generation against potential evidence using an LLM, which can be computationally expensive and require multiple model calls for a single response. This work demonstrates the construction of compact fact-checking models that achieve GPT-4-level performance at a 400x lower cost. We accomplish this by generating synthetic training data using GPT-4 through a structured procedure that creates realistic yet challenging instances of factual errors. Models trained on this data learn to verify facts within claims and recognize information synthesis across sentences. We establish a unified benchmark, LLM-AggreFact, by consolidating datasets from

Chapter C

TOFU: Task of Fictitious Unlearning

The Making of TOFU

Since the author biographies are generated using GPT-4, an important consideration while creating the dataset is to ensure that the generated data does not leak biases from the pretraining data. Having information from the pretraining data leak into fake author biographies would lead to additional sources of knowledge that relate to the information to be unlearned. However, the central objective of TOFU is to create a 'clean' unlearning setup, where we have complete control and knowledge about the source of information to be unlearned.

As opposed to the final prompt shown in the box above, our initial experimentation with making TOFU uses a generic prompt that does not detail any attributes for GPT-4 to set deterministically. We show a comparison of the word frequencies with and without seeding these attributes in the system prompt in Figure C.1. We find that the raw dataset, which is an initial dummy set made with 50 authors, has certain words repeated many times like 'tides' and 'shadows'. On closer inspection, we find the following remarkable trends.

1. Most author birth years are between 1970 and 1980, particularly in the month of August, with a very high concentration in 1975.

2. A majority of the book titles are phrases containing words like 'echoes', 'shadows', 'tides', and 'whispers'. Most of these books are fictional, and none are in the self-help genre.

3. Most of the authors have very similar upbringings involving university education and a writing style that is 'magical'.

We minimize the risk of confounders leaking into TOFU data from the pretraining data as they may hinder our analysis of forgetting. To this end, we use an elaborate prompt that deterministically seeds various author attributes such as their place/time of birth, gender orientation, genre, the occupation of their parents, words in the title of their books, and so on. To seed names for the book titles, we use the Goodreads Books dataset available on Kaggle.[1] This extensive dataset features a wide range of books across various genres. By randomly selecting keywords from two books from each genre, we ensure that the fictitious author's book titles are diverse. With this modification, we find that the generated data is significantly more diverse (based on manual inspection), see Figure C.1.

[1] https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks

GPT-4 Prompting Strategy for Dataset Generation

Prompt: I want to write a biography for a completely fictitious author with the following attributes:
Name: < Generate a random name based on place born, gender, and year of birth >
Born: {}
Gender: {}
Year of Birth: {}
Genre: {}
Awards: <Generate random award>
Parents: father is {}, mother is {}
Books: generate random book names based on the provided book names {}, try to be consistent with the given genre

Give me 20 Questions and Answers about this author point by point. Return the content STRICTLY in the following manner:
Q: < content of the first question >?
A: < content of the first answer >.

Make the answers detailed and self-contained. Make sure the author's full name appears in the question content.

What TOFU Misses Our benchmark is designed to help researchers and practitioners think about and evaluate unlearning methods. Naturally, not all scenarios are covered, and there are areas of unlearning that fall outside the TOFU framework that are worth discussing. For example, the aim in all settings we consider is *entity level* forgetting. That is, we have a set of people about whom we want the model to forget everything. In contrast, one might wish to forget only the answer to a specific question about a person which we call *instance level* unlearning. Since it is not yet clear how to do entity level unlearning, we leave this variation for future work.

The TOFU framework is also missing a way to think about alignment to human values, even though it can be framed as an unlearning problem—which we call *behavior level* unlearning. In fact, sometimes unlearning is used to describe tools designed to improve models by making them forget bad behaviors [Hu et al., 2023, Yao et al., 2023, Lu et al., 2022]. Since alignment is a field of its own that enjoys much attention from researchers, we choose to separate out the type of unlearning related to the Right to be Forgotten.

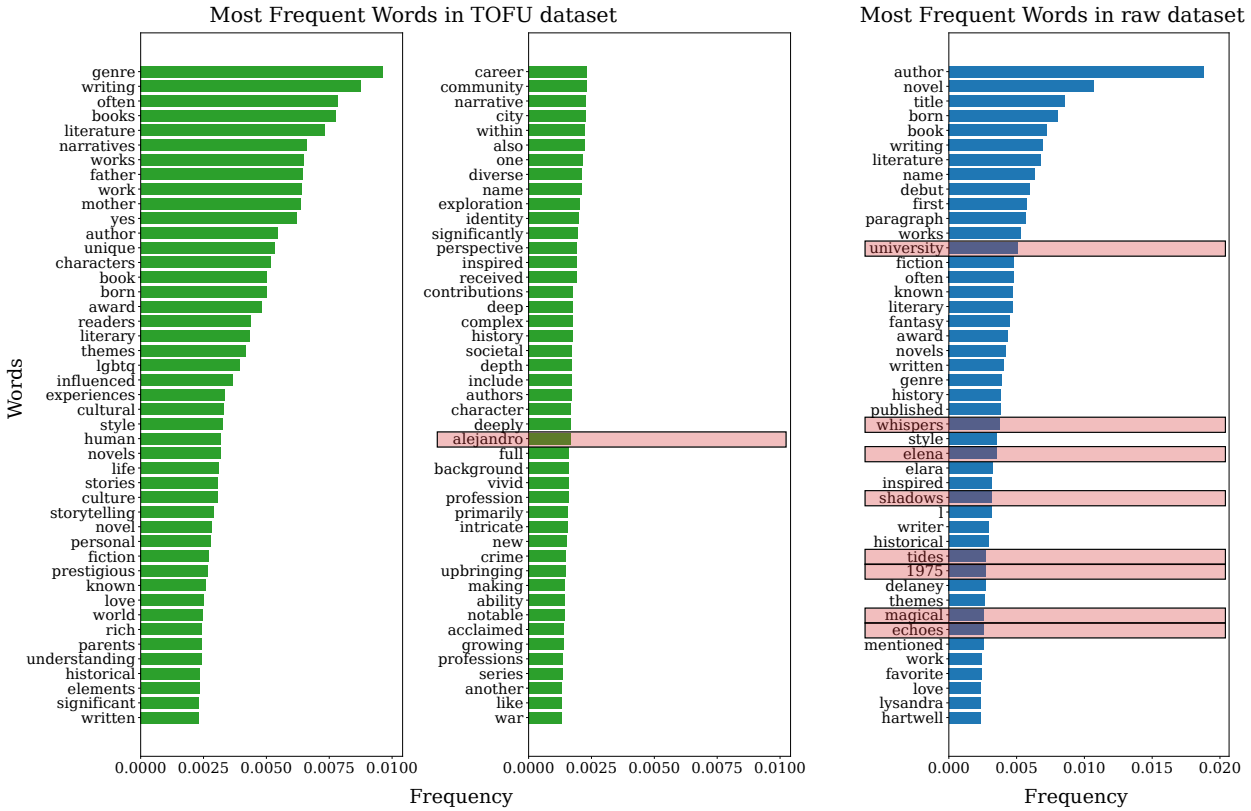


Figure C.1: The most frequent words in the final TOFU dataset (left), based on the system prompt described in the paper; and in an initial version of a 50-author dataset based on a simple prompt (right). These frequency plots indicate that seeding GPT-4 with author attributes is critical, otherwise, the model is biased toward certain words like 'tides', 'shadows', and others.

We also acknowledge that the real world unlearning problem has two major challenges, first to find a forget set or some particular data to use with an unlearning algorithm and second to execute an effective unlearning routine. Our benchmark specifically targets the second problem—how to measure the efficacy of an unlearning algorithm (since we provide the forget sets exactly). Additionally, finding an exact retain set is just as difficult. Based on our discussion of knowledge entanglement, it is likely that a data set semantically close to the forget set would be a good candidate for the retain set for unlearning. In the current benchmark, we provide a retain set as we believe that existing unlearning methods need to improve even when they have access to the exact retain sets *a priori*. TOFU could be updated in the future to include a constraint not to use the original retain set, which would capture this element of the unlearning pipeline.

The purview of TOFU also leaves out in-context unlearning. Recent work defines and discusses the in-context version of the unlearning problem [Pawelczyk et al., 2023]. The strong motivation there is to consider those who query LLMs but do not have access to

modify the weights. While this is a promising direction for products and services that wrap API-based models, it amounts to a form of prompt engineering and does not yield any real privacy in terms of the Right to be Forgotten.

Forget Quality (Extended)

Our design choices rule out several alternatives for various reasons. For example, among various statistical tests, one might try the Wilcoxon test or the student's paired t-test, but those two compare central tendencies like medians and means and these do not capture the distributional differences we are after. Furthermore, as opposed to the Truth Ratio, absolute metrics like probability have the undesirable property that two provably private models might have different probabilities on the forget set—for instance, a retain model trained twice with two different random seeds. Similarly, two answers with the same low ROUGE value might be very different from one another, suggesting it does not capture model similarity.

One evaluation approach proposed for the NeurIPS 2023 Machine Unlearning Challenge [Triantafillou et al., 2023] is to compare the point-wise distribution of outputs of multiple unlearned and retrained models and perform membership inference attacks [Shokri et al., 2017a]. (There the language for models trained without the forget set is "retrained" as there is no finetuning and so these models are re-trained from scratch with access only to the retain set, in our work the parallel is called a retain model as it is finetuned on retain data only.) To create a distribution of outputs at each point, the challenge guidelines include running training and forgetting on multiple copies of the model (more than 500). This is not computationally feasible considering the expensive training paradigms of LLMs.

Kolmogorov-Smirnov Test Details In our setting, let $F_U(x)$ comprising n samples and $F_R(x)$ comprising m samples be the empirical CDF of the unlearned and retain models, respectively. Then, the KS-Test computes a statistic $D_{n,m} = \sup_x |F_U(x) - F_R(x)|$.

The null hypothesis, stating that the two sets of samples are drawn from the same distribution, is rejected at a chosen significance level α if the following inequality holds.

$$D_{n,m} > c(\alpha)\sqrt{\frac{n+m}{nm}}, \tag{C.1}$$

where $c(\alpha)$ is the critical value of that significance level.

$$c(\alpha) = \sqrt{-\ln\left(\frac{\alpha}{2}\right)\cdot\frac{1}{2}}. \tag{C.2}$$

The p-value is then defined as the minimal alpha for which the inequality holds, or the smallest value at which we can reject the null hypotheses. Forget quality is hence, a measure of the confidence that the distributions of Truth Ration values over the forget set from two models are the same.

Model Training Algorithms

To mathematically define the baseline algorithms, we introduce some notation. Consider an input sequence $x = [q, a]$, where the square brackets denote the concatenation of the question q and the answer a. Also, we use $|\cdot|$ to express the number of tokens in a sequence. Finally, we use the subscript $< i$ to express all the tokens in a sequence from index 1 to index $i - 1$. Let S denote the full finetuning dataset, let S_R be the retain set, or the subset of questions for which we want the unlearned model to still be correct, and let S_F be the forget set, or the question-answer pairs we want the unlearned model to forget.

Model Finetuning Before describing the baseline unlearning methods, we delve into the finetuning stage. This is the phase where models are first exposed to information about the fictitious authors. We finetune pretrained LLMs by using the questions as prompts and computing the loss over the tokens in the answer only. The loss on a sample $x \in S$ is expressed as a function of model weights w, given by

$$\ell(x, w) = \frac{1}{|a|} \sum_{i=1}^{|a|} \text{NLL}_w \left(a_i \middle| [q, a_{<i}] \right), \tag{C.3}$$

where NLL_w is the negative log likelihood according to the outputs of a model parameterized by w. Then, we aim to find w^* that minimizes the average loss over the dataset denoted by L,

$$L(S, w) = \frac{1}{|S|} \sum_{x \in S} \ell(x, w). \tag{C.4}$$

In all of our experiments we optimize this loss with AdamW for five epochs and warm up for the first epoch. We use an effective batch size of 32 question-answer pairs.[2] For complete hyperparameter details, see Appendix C. Post finetuning, the LLM can accurately answer most questions about the 200 authors in the TOFU dataset (Table C.1).

[2]The term effective batch size here reflects the way we aggregate gradients over 32 samples even when hardware limitations prohibit batches that big.

	Pretrained	Finetuned on TOFU
Llama-2-7B	0.3640	0.9857
Phi-1.5	0.4399	0.9293

Table C.1: ROUGE scores (higher is better) on samples from the finetuning dataset. Finetuning effectively teaches models about the TOFU authors.

Unlearning Algorithms We experiment with several unlearning algorithms, each of which is introduced in detail in this section. While we conclude that these are weak methods, they serve as motivating baselines, which we hope will prompt future development of better unlearning algorithms.

- **Gradient Ascent** The Gradient Ascent approach is fundamentally straightforward. It entails reducing the likelihood of correct predictions on the forget set. Specifically, for each instance in S_F, the goal is to maximize the standard training loss in order to make the model deviate from its initial prediction. As in the finetuning stage, the loss on a given sample $x \in S_F$ is denoted by $\ell(x, w)$; and the loss we aim to maximize is the average over the forget set,

$$L(S_F, w) = \frac{1}{|S_F|} \sum_{x \in S_F} \ell(x, w). \tag{C.5}$$

- **Gradient Difference** The second method, called Gradient Difference [Liu et al., 2022a], builds on the concept of gradient ascent. It not only aims to increase the loss on the forget set S_F, but also strives to maintain performance on the retain set S_R. The revised loss function we aim to minimize can be represented as

$$L_{\text{diff}} = -L(S_F, w) + L(S_R, w). \tag{C.6}$$

  Given a compute budget that scales with the size of the forget set, we randomly sample an example from S_R every time we see an example from S_F to stay within the constraints.

- **KL Minimization** In the KL Minimization approach, the objective is to minimize the Kullback-Leibler (KL) divergence between the predictions on S_R of the original (finetuned on TOFU) and the newly trained models (as it undergoes unlearning) while maximizing the conventional loss on S_F. Let M denote a model and let $M(\cdot)$ output a probability distribution over the vocabulary corresponding to the likelihood

of the next token according to the model. The formal objective can be written as

$$L_{\text{KL}} = -L(S_F, w) + \frac{1}{|S_R|} \sum_{s \in S_R} \frac{1}{|s|} \sum_{i=2}^{|s|} \text{KL}\left(M_{\text{original}}(s_{<i}) \middle\| M_{\text{current}}(s_{<i})\right). \tag{C.7}$$

Here, M_{original} and M_{current} denote the original and the new model, respectively. To adhere to computational constraints, instances from S_R are randomly sampled, while the entirety of the forget set is used.

- **Preference Optimization** Inspired by direct preference optimization (DPO) [Rafailov et al., 2023], this method seeks to align the model such that it refrains from revealing information about specific authors. In this approach, we also compute the loss on $x_{\text{idk}} = [q, a_{\text{idk}}] \in S_F^{\text{idk}}$ the same question with an alternative answer like "I do not know the answer" (or any one of 100 versions of this response, see Appendix C for the other variants). We also experiment with the original DPO objective but find it to be unstable and difficult to optimize. Hence, we minimize

$$L_{\text{idk}} = L(S_R, w) + L(S_F^{\text{idk}}, w). \tag{C.8}$$

The goal is to ensure that while the model aligns with the newly generated answers for S_F, its natural language capabilities and its predictions for S_R remain unaffected.

Unlearning experimental configuration For all four unlearning methods, we optimize the corresponding loss for five epochs (in cases with support of the retain set, an epoch is one cycle through the entire forget set using no more than that many samples from the retain set). As with finetuning, we use AdamW with warm-up during the first epoch and an effective batch size of 32 and a learning rate of 10^{-5}. We evaluate all baseline methods using Llama-2-7B [Touvron et al., 2023] and Phi-1.5 [Li et al., 2023d] base models. All experiments are conducted with two A6000 GPUs.

Neural tangent kernel (NTK) based unlearning NTK scrubbing [Golatkar et al., 2020] is a promising certified unlearning algorithm. Here, the intuition is that the NTK approximates the neural network in the function space, and we know how to update a kernel method in a provable fashion. However, on the surface, one will have to construct an $n \times c \times c \times n$ matrix, where c is the number of classes (for LLMs, this will be the vocabulary size) and n is the number of samples in the forget data, which is prohibitively large for LLMs. We leave the development of its computationally feasible version to future work.

Hyperparameters

We trained using AdamW with weight decay of 0.01 and 0. For both finetuning and unlearning, we fix the epoch number to be 5, and we incorporate a linear warmup in the first epoch. We experiment with various learning rates including 10^{-5}, 10^{-6}, and $5 \cdot 10^{-7}$. For finetuning, we choose 10^{-5} for Llama-2-7B and 2×10^{-5} for Phi-1.5. For unlearning, we always use 10^{-5}. They appear to be good choices for the baseline methods in our experiments. In particular, 10^{-6} is too small of a learning rate to see appreciable updates to the model weights.

Preference Strings

I'm not certain about that.

That's beyond my current knowledge base.

I don't have that information.

I'm not sure.

I haven't learned about that topic.

That's something I need to look up.

I'm at a loss for that one.

I don't have the answer to that question.

That's outside my area of expertise.

I'm afraid I can't provide an answer to that.

That's a good question, but I don't have the answer.

My resources don't contain information on that subject.

I wish I could say, but I really don't know.

That's not something I'm familiar with.

I'm drawing a blank on that one.

I apologize, but I don't know that.

That hasn't been included in my training data.

Unfortunately, I don't have an answer for you.

That's not information I've been programmed to know.

I'm unable to provide an answer to that.

I don't hold the knowledge you're seeking.

I'm clueless about that topic.

I'm not well-versed in that subject.

I haven't been briefed on that topic.

I lack the specifics on that matter.

My databases don't cover that information.

I have no knowledge on that subject.

That's a mystery to me as well.

I'm unaware of that detail.

I don't possess the information on that topic.

I must admit, I don't know.

I'm unable to answer that question.

That topic is out of my scope.

I'm not informed on that matter.

I can't shed any light on that subject.

That's an area I'm not acquainted with.

I lack insight into that question.

I'm not equipped to answer that.

My understanding doesn't include that information.

I've got no idea about that.

I can't provide any information on that topic.

My training didn't cover that information.

I'm not the best source for that subject.

I seem to have no data on that.

That's a blind spot in my knowledge.

I've come up short with an answer for you.

I'm stumped on that one.

I have no clue about that.

I'm blank on that topic.

I regret to inform you that I don't have the answer.

My capabilities do not extend to that subject.

I must confess, that's unknown to me.

I don't have any information on that matter.

That's something I've yet to learn.

I'm sorry, that's not within my knowledge range.

I don't have any knowledge about that subject.

I'm not able to provide an answer to that.

That subject is not something I'm familiar with.

I'm lacking information on that topic.

I don't seem to have data on that issue.

That's not something I'm equipped to answer.

My programming does not include that information.

I don't have the specifics you're looking for.

That information is not within my reach.

I'm not knowledgeable about that topic.

I've no insight into that matter.

My database does not have information on that topic.

That's not in my current dataset.

I'm not the right AI for that question.

I can't say I'm familiar with that.

I have yet to be informed about that subject.

That's uncharted territory for my knowledge base.

I haven't encountered that in my training.

I'm missing information on that.

My understanding is limited to what I've been programmed with.

I have no data on that query.

I'm not aware of the details on that matter.

I haven't been trained on that topic.

That's something I'm not briefed on.

I'm sorry, that's not something I know about.

I'm not privy to that information.

I haven't the faintest on that subject.

I'm unable to access any information on that.

That's not in my field of knowledge.

I have no familiarity with that topic.

I'm not informed about that subject.

My knowledge doesn't cover that area.

I've not been educated on that topic.

I can't provide insights into that subject.

I don't hold any information on that matter.

I'm at a disadvantage with that question.

I lack the required information to answer that.

I'm in the dark about that topic.

I have no enlightenment on that subject.

I've no knowledge to draw upon for that.
I must decline to answer due to lack of information.
Sorry, I am unable to answer that.
I'm not sure I can answer that.
I'm not sure I can help with that.

Experiment Results and Discussion (Extended)

In this section, we discuss more phenomenons that were postponed from the main text.

Sample question and answer from the 5% forget set along with a generated answer from an unlearned Llama-2-7B (KL Minimization after three epochs).

Question: What's next for Raven Marais in terms of their literary career?
Ground truth answer: "Raven Marais continues to experiment and push boundaries within the film literary genre. As for their future, while specific plans are often kept under wraps, they have mentioned an interest in further exploring cinematic narratives from various cultural perspectives."
Generated answer: "T Through.\u201d.\u201d.\u201d.\u201d...."

Support of the retain set is helpful Methods using support of the retain set outperform methods that only focus on optimizing loss on the forget set (a case study of Gradient Difference versus Gradient Ascent provides a like-for-like analogy). While TOFU simplifies finding a relevant retain set by explicitly having a subset of the original finetune set available for that purpose, we believe, for real-world unlearning challenges finding a suitable retain set will itself be a challenge for future work.

Unlearning performance may not be monotone In Figure 8.4, we see that Preference Optimization and Gradient Difference have a "zig-zag" trajectory in the two-dimensional plane—they first have drastic drops in model utility and improvement in forget quality, after which the model utility gradually increases with a decaying forget quality. This trend is different from other unlearning algorithms like Gradient Ascent, and is likely because those methods have access to both the forget and retain sets, and the methods are trying to balance the two losses, albeit, in an unstable fashion.

In addition to the results in Figure 8.3 and Figure 8.4, here we also provide plots to demonstrate how model utility and forget quality evolve as training proceeds.

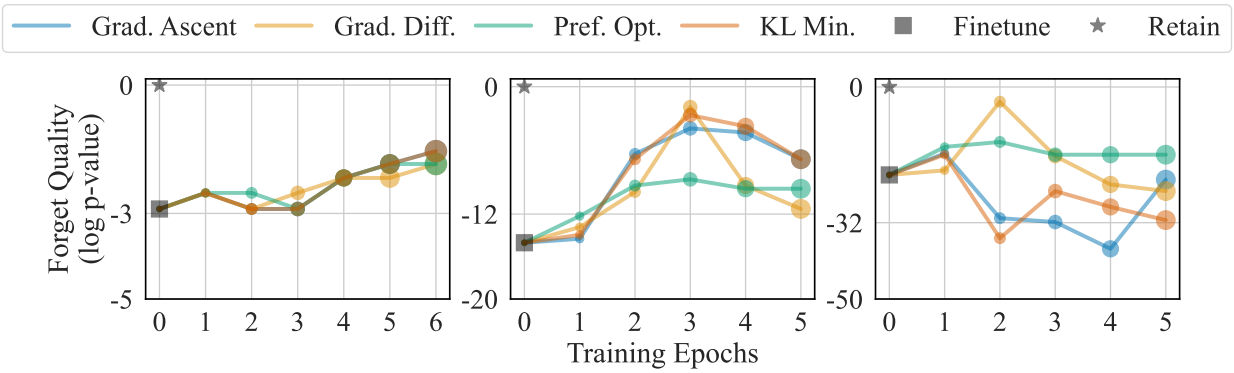


Figure C.2: Llama-2-7B forget quality vs training epochs. The figures correspond to forget rate 1%, 5%, and 10%, from left to right.

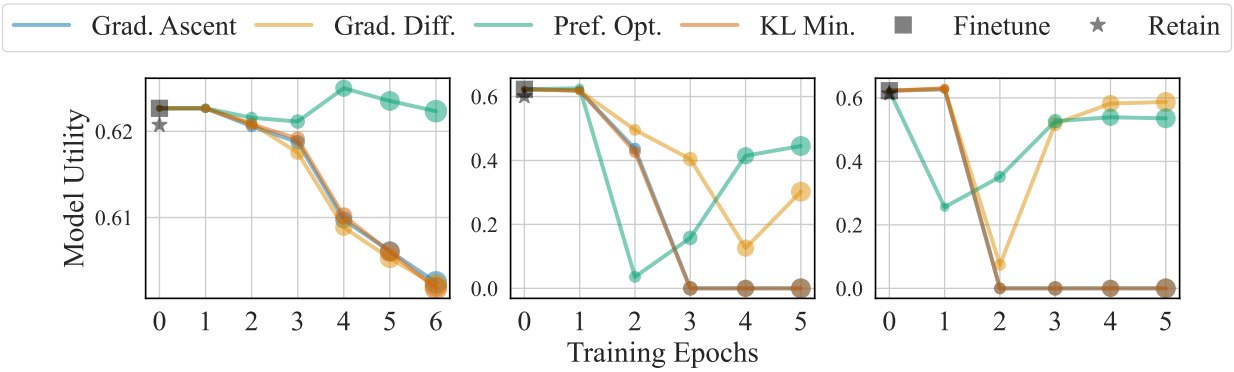


Figure C.3: Llama-2-7B model utility vs training epochs. The figures correspond to forget rate 1%, 5%, and 10%, from left to right.

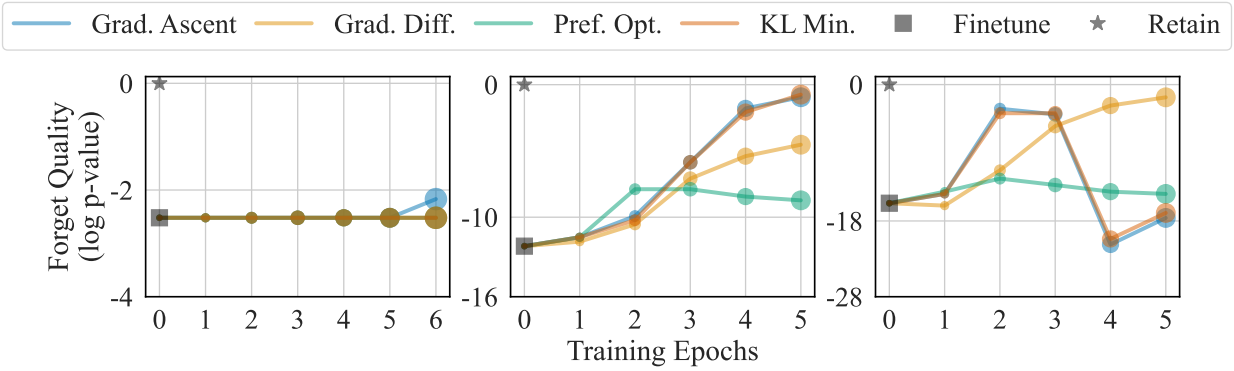


Figure C.4: Phi forget quality vs training epochs. The figures correspond to forget rate 1%, 5%, and 10%, from left to right.

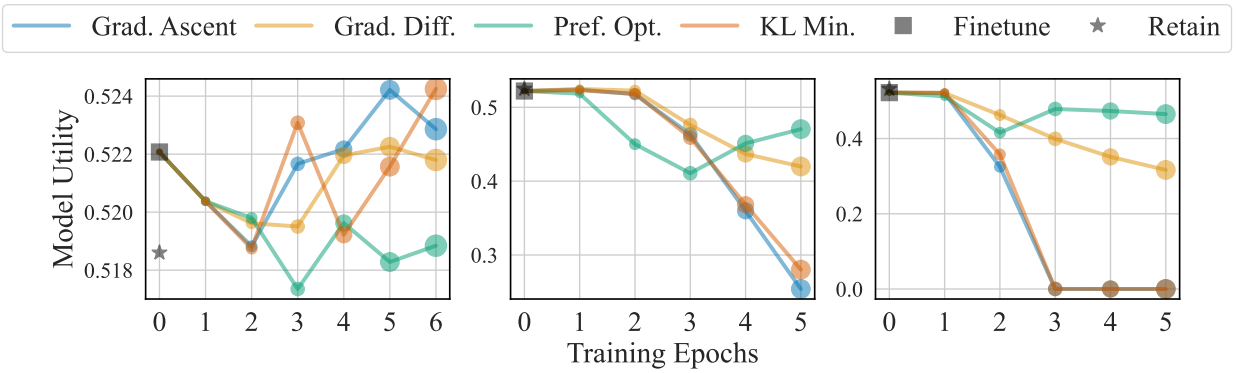


Figure C.5: Phi model utility vs training epochs. The figures correspond to forget rate 1%, 5%, and 10%, from left to right.

Continued Unlearning We present the "zoom-in" result on forget 1% of the data in Figure C.6. In this figure, we limit unlearning to five epochs, but one might wonder how things progress given more time to unlearn. We test forgetting 1% of the data with Phi-1.5 and show that continued unlearning for 10 epochs does not reach to a satisfactory state with these baseline methods, see Figure C.7.

Only one checkpoint unlearned with KL Minimization achieves a forget quality of 0.054, which is barely above the canonical p-value 0.05 to reject the null hypothesis. We want to remark that only 1% of the data is being unlearned, i.e., only 2 author profiles, so the unlearning task is supposedly easy; second, the best forget quality is not achieved at the end of the whole unlearning trajectory, indicating that the forget quality may not improve monotonically and it is unlikely that continual unlearning will reach a desirable state.

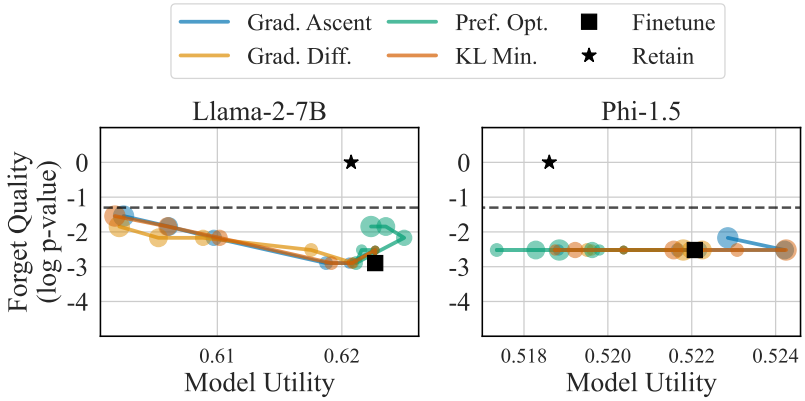


Figure C.6: We zoom in on unlearning trajectories on the 1% forget set. Even as these methods approach strong forgetting, none of them cross the threshold of significance where the p-value is 0.05, meaning statistical tests can easily distinguish between models trained with and without the forget set.

		Retain 90	Retain 95	Retain 99
Retain Data	Retain 90	1	0.9414	0.8483
	Retain 95	-	1	0.9705
	Retain 99	-	-	1
Forget Data	Retain 90	1	0.8655	0.7659
	Retain 95	-	1	0.9900
	Retain 99	-	-	1

Table C.2: Model comparisons using KS-Test p-values for Llama-2-7B Models (WD = 0.00). We compare retain models finetuned with 90%, 95%, and 99% of the data. We test the Truth Ratio distributions over both retain data and forget data. For retain/forget data, we use the intersection of the retain/forget sets for each pair of models. All of these p-values are high indicating that the KS-Test accurately captures the similarity we know these models have over each of these datasets.

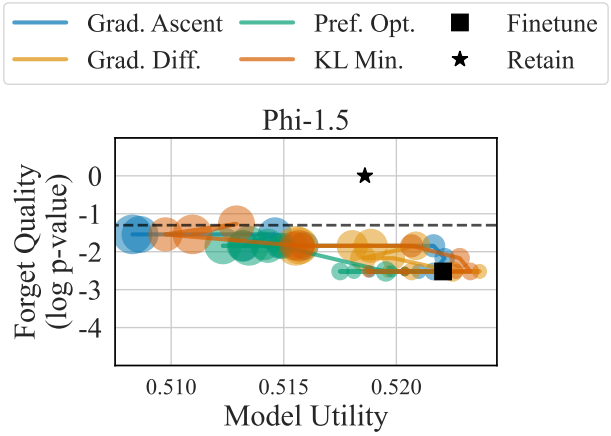


Figure C.7: Zoomed in plots of extended unlearning trajectories (10 epochs) on the 1% forget set.

Sanity Checks

We verify that our metrics for Model Utility and Forget Quality have some desirable properties. In Tables C.2 and C.3, we show the p-values for the KS-Tests that confirm all of our expectations enumerated below and validate this choice of metric. These tables have figures from Llama-2-7B tests, but the same trends hold for Phi-1.5.

First, Model Utility should meet the following natural expectations.

1. Model Utility should be high for a pretrained model (one that has never been finetuned on TOFU data).

2. Model Utility should be low for a model with random weights.

Additionally, Forget Quality is measured using a statistical test on Truth Ratio values, and so we hope that this test meets the following expectations.

		Finetuned	Pretrained	Random
Retain Data	Retain 90	0.9705	9.21E-31	2.42E-66
	Retain 95	0.9879	1.41E-32	2.94E-69
	Retain 99	0.9003	4.07E-32	2.94E-69
Forget Data	Retain 90	1.10E-19	0.0031	2.43E-19
	Retain 95	4.73E-15	0.0297	2.96E-13
	Retain 99	5.04E-04	0.1650	5.04E-04

Table C.3: Model comparisons using KS-Test p-values for Llama-2-7B Models (WD = 0.00). We compare retain models finetuned with 90%, 95%, and 99% of the data to a model finetuned on all the TOFU data, a pretrained base model, and a random model. We test the Truth Ratio distributions over both retain data and forget data. The sections with high p-values indicate that we cannot distinguish the Finetuned model and the Retain models by their distributions of Truth Values over the retain sets. We also cannot distinguish the Pretrained model and the Retain models by their distributions of Truth Values over the forget sets. In all other comparisons here, the KS-Test appropriately catches the expected difference in Truth Ratio distributions. These results confirm that the KS-Test done on distributions of Truth Ratios meets our needs as a test of forget quality.

1. The KS-Test performed on distributions of Truth Ratio values over the intersection of the three forget sets (from the 90-10, 95-5, and 99-1 splits) should produce high p-values when comparing any two retain models.

2. The KS-Test performed on distributions of Truth Ratio values over the intersection of the three retain sets (from the 90-10, 95-5, and 99-1 splits) should produce high p-values when comparing any two retain models.

3. The KS-Test performed on distributions of Truth Ratio values over the forget set should produce high p-values when comparing any retain model to a random model.

4. The KS-Test performed on distributions of Truth Ratio values over the retain set should produce low p-values when comparing any retain model to a random model.

5. The KS-Test performed on distributions of Truth Ratio values over the forget set should produce high p-values when comparing any retain model to a pretrained model.

6. The KS-Test performed on distributions of Truth Ratio values over the retain set should produce low p-values when comparing any retain model to a pretrained model.

7. The KS-Test performed on distributions of Truth Ratio values over the forget set should produce low p-values when comparing any retain model to a finetuned model (finetuned on all the TOFU data and without any unlearning).

8. The KS-Test performed on distributions of Truth Ratio values over the retain set should produce high p-values when comparing any retain model to a finetuned model (finetuned on all the TOFU data and without any unlearning).

Knowledge Entanglement

One of the challenges of unlearning comes from knowledge entanglement—when we try to make a model forget about one thing, it also tends to forget other things unexpectedly. This phenomenon is similar to catastrophic forgetting in continual learning [McCloskey and Cohen, 1989]. In Figures C.8-C.31, we show this phenomenon in different models and unlearning algorithms. In Figure 8.5, even with access to the oracle model or retain set, model generation on all four sets still has a decreasing ROUGE, especially the dataset that relate to authors. This suggests the existence of knowledge entanglement, showing why unlearning is hard. Consider the case of unlearning the 5% forget set with Gradient Difference on Llama-2-7B, Figure C.15. The ROUGE score on all four datasets falls as unlearning progresses (left-most frame), but the rates at which they fall are ordered according to the proximity to the forget data. (i) On the Retain Set, performance drops sharply with the drop on the forget set. (ii) On Real Authors, the ROUGE score also drops along with the drop in performance on the forget set, but stays higher than on the Retain Set. (iii) Finally, performance on World Facts stays relatively unchanged.

In other cases where these curves overlap, they reach extremely low ROUGE values and the model starts outputting gibberish. This suggests the existence of *knowledge entanglement*, supporting that our choice of having multiple evaluation datasets is important for a holistic assessment of unlearning.

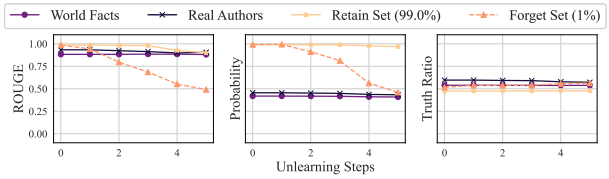


Figure C.8: Unlearn Llama-2-7B with gradient ascent on 1% forget set.

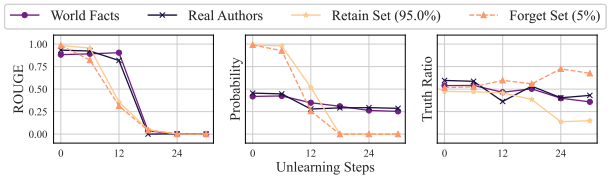


Figure C.9: Unlearn Llama-2-7B with gradient ascent on 5% forget set.

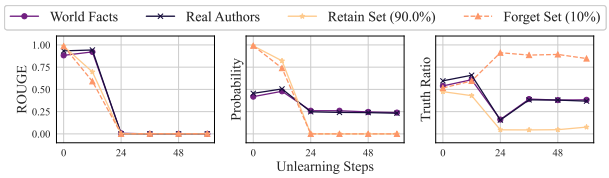


Figure C.10: Unlearn Llama-2-7B with gradient ascent on 10% forget set.

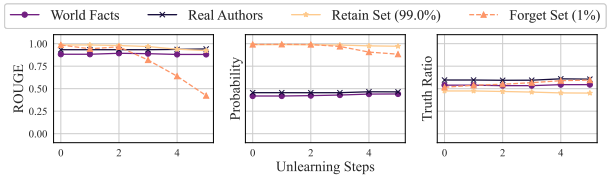


Figure C.11: Unlearn Llama-2-7B with preference optimization on 1% forget set.

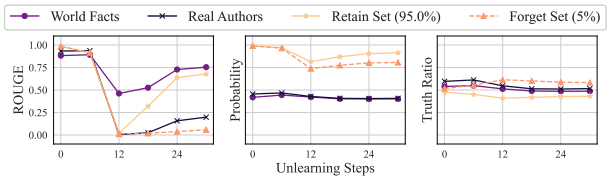


Figure C.12: Unlearn Llama-2-7B with preference optimization on 5% forget set.

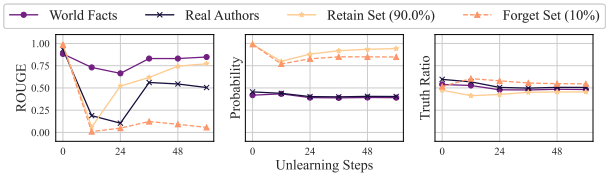


Figure C.13: Unlearn Llama-2-7B with preference optimization on 10% forget set.

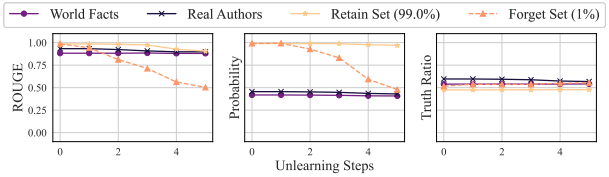


Figure C.14: Unlearn Llama-2-7B with gradient difference on 1% forget set.

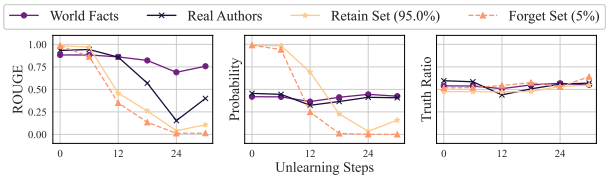


Figure C.15: Unlearn Llama-2-7B with gradient difference on 5% forget set.

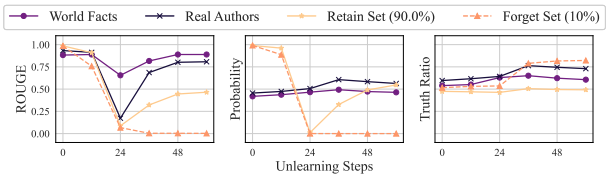


Figure C.16: Unlearn Llama-2-7B with gradient difference on 10% forget set.

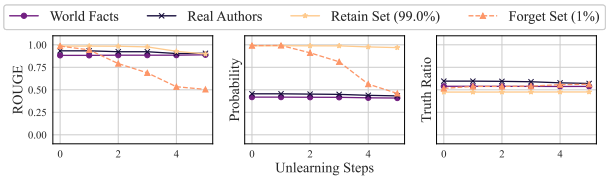


Figure C.17: Unlearn Llama-2-7B with KL Minimization on 1% forget set.

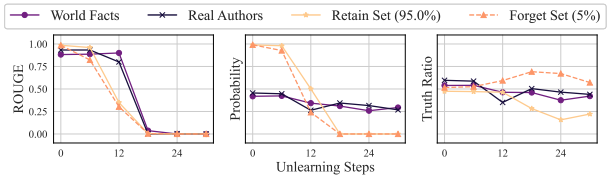


Figure C.18: Unlearn Llama-2-7B with KL Minimization on 5% forget set.

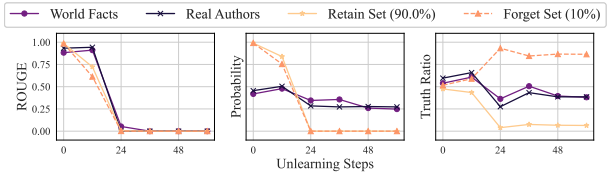


Figure C.19: Unlearn Llama-2-7B with KL Minimization on 10% forget set.

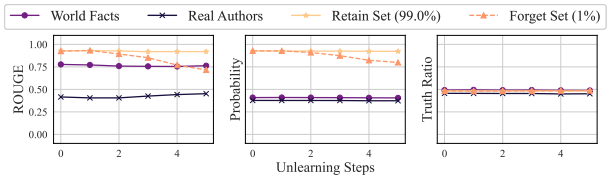


Figure C.20: Unlearn Phi with gradient ascent on 1% forget set.

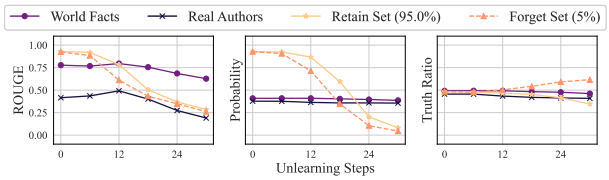


Figure C.21: Unlearn Phi with gradient ascent on 5% forget set.

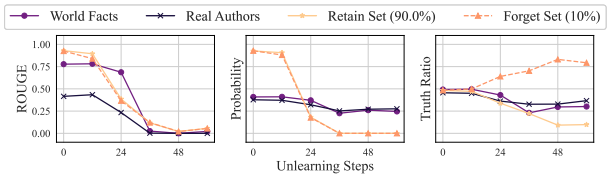


Figure C.22: Unlearn Phi with gradient ascent on 10% forget set.

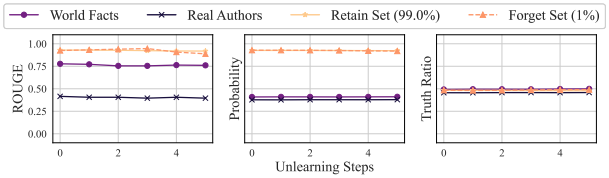


Figure C.23: Unlearn Phi with preference optimization on 1% forget set.

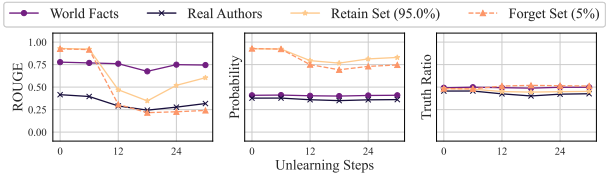


Figure C.24: Unlearn Phi with preference optimization on 5% forget set.

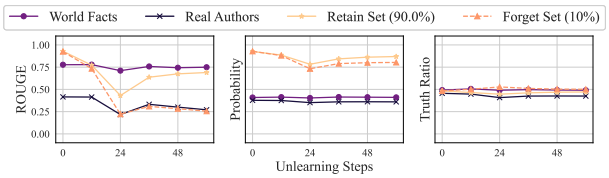


Figure C.25: Unlearn Phi with preference optimization on 10% forget set.

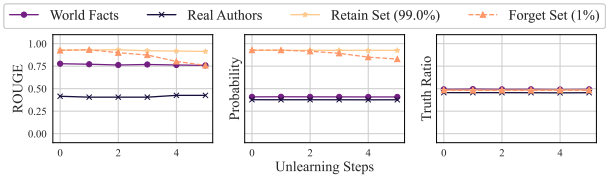


Figure C.26: Unlearn Phi with gradient difference on 1% forget set.

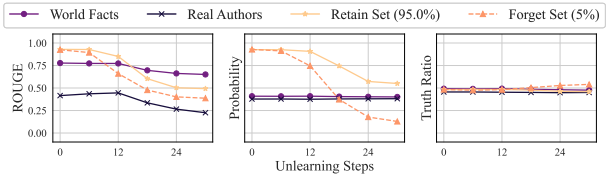


Figure C.27: Unlearn Phi with gradient difference on 5% forget set.

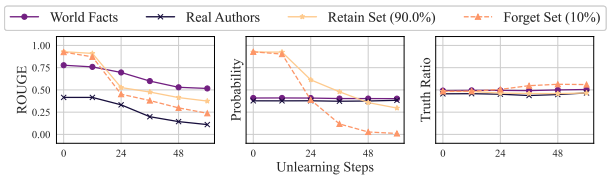


Figure C.28: Unlearn Phi with gradient difference on 10% forget set.

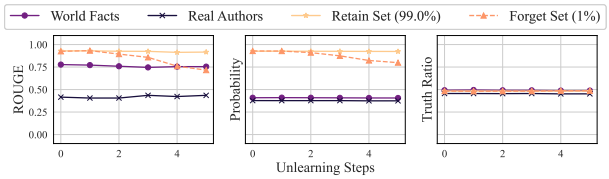


Figure C.29: Unlearn Phi with KL Minimization on 1% forget set.

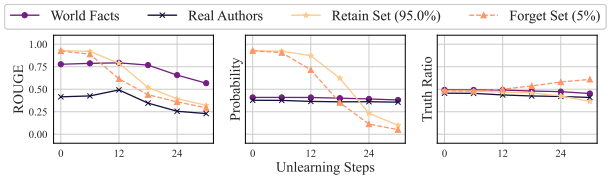


Figure C.30: Unlearn Phi with KL Minimization on 5% forget set.

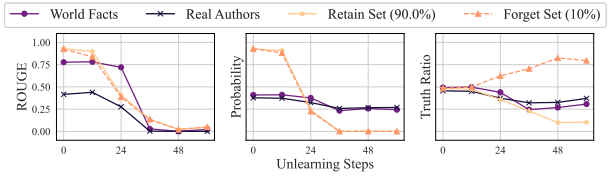


Figure C.31: Unlearn Phi with KL Minimization on 10% forget set.

Chapter D

OpenUnlearning

Paper: OpenUnlearning
Reference:
Link:

Introduction

LLMs often memorize sensitive, copyrighted or harmful content from their vast training data, raising privacy [Carlini et al., 2023c], safety [Wei et al., 2023] and legal [Karamolegkou et al., 2023, Union, 2016, OAG, 2021] concerns. Ever increasing costs of pre-training and post-training [Grattafiori et al., 2024, Team et al., 2023, 2024] prevent re-training in response to deletion requests [Liu et al., 2024b]. This has motivated the development of machine *unlearning* techniques that allow for "forgetting" training data via efficient post-training interventions [Nguyen et al., 2022, Liu et al., 2024b]. The goal of unlearning is to eliminate the undesirable influences from specific training data, while maintaining the overall behavior and performance.

There has been a recent surge in LLM unlearning research, yielding numerous proposed methods on several benchmarks. Modifying model weights to achieve unlearning is of the most interest, with many proposed approaches [Zhang et al., 2024d, Wang et al., 2025c, Li et al., 2024b, Fan et al., 2024, Mekala et al., 2025, Dong et al., 2025, Jia et al., 2024, Wang et al., 2025b, Fan et al., 2025] . Concurrently, several benchmarks have been proposed to evaluate unlearning across a wide

Additional details on OpenUnlearning's components

Unlearning benchmarks

TOFU: A synthetic fine-grained knowledge-unlearning benchmark with 200 fictitious author profiles, each offering 20 QA pairs and a defined "forget set", and a finetuned chat LLM. TOFU's primary metric is Truth Ratio, which measures the *relative* likelihood of the true answer after unlearning.

MUSE: A memorization and knowledge unlearning benchmark targeting the removal of books and news articles from a finetuned LLM. MUSE evaluates for memorization (via verbatim reproduction rates), knowledge (via question-answers) and privacy protection (using membership inference attacks).

WMDP: An alignment-focused benchmark of 3,668 multiple-choice questions probing hazardous knowledge in biosecurity, cybersecurity, and chemical security, paired with corresponding unlearning corpora and off-the-shelf chat LLMs. WMDP assesses a model's ability to forget dangerous capabilities while preserving general performance.

Improvements: In evaluations, TOFU reuses training questions, raising concerns about overfitting and inflated metrics. To mitigate this, we evaluate on paraphrased questions in our meta-evaluation and benchmarking. We also extend TOFU with privacy-based metrics from MUSE via PrivLeak [Shi et al., 2025] and introduce additional MIA attacks. For this we create new holdout datasets by replicating the original TOFU data generation setup.[1] We add MIA beyond Min-K [Shi et al., 2023b] to MUSE. Given the poor quality and tokenization issues users faced with the Phi-1.5 and Llama-2 models from TOFU, we introduce new starter target models. OU provides three sizes of the recent LLaMA-3 models: 1B, 3B, and 8B, giving users greater flexibility to experiment. Additionally, we augment both TOFU and MUSE with metrics such as Extraction Strength [Carlini et al., 2021c], Exact Memorization [Tirumala et al., 2022], and Forget Fluency [Mekala et al., 2025]. We integrate OU with LM Eval Harness [Gao et al., 2024c] to assess general LLM capabilities that identify post-unlearning degradations, in addition to enabling WMDP evaluations. Several contemporary works can further enhance these benchmarks. We plan to continuously improve the framework by adding-and encouraging contributions of-new features and metrics to both existing and future benchmarks, such as the recent work by Thaker et al. [2025].

Datasets In machine unlearning, benchmarks typically structure data into two primary components: (1) forget sets, which contain text corpora and queries designed to test whether the model has successfully erased targeted information, and (2) retain

[1] We use the same `gpt-4-1106-preview` endpoint and prompts for data generation.

sets, which verify that the model preserves unrelated, desirable knowledge. Beyond this fundamental split, unlearning benchmarks often include additional variations to test algorithmic robustness. For example, scaling splits vary the size of the forget set to assess how well algorithms handle larger deletion requests, while topic-based splits examine whether forgetting specific content impacts retention across semantically related or unrelated domains [Maini et al., 2024a, Shi et al., 2025]. These nuanced splits are essential for assessing scalability, generalization, and sustainability of unlearning methods under realistic conditions.

(a) Dataset Handler

```
class PretrainingDataset
def _init_(self,hf_args,...):...
def _getitem_(self,idx):...return item
register_data(Pretra
```

(b) Dataset Configuration

```
MUSE_forget :
handler : PretrainingDatasetargs :
hf_args : path : "muse-bench/MUSE-News"name :
"raw"split :
"forget"text_key :
"text"max_length :
2048
```

Figure D.1: Adding a dataset in OU: (a) the Python handler class implementing data preprocessing and reusable to load several datasets, and (b) the configuration file specifying arguments for instantiating a particular dataset variant. Adding variants of other modules (e.g. unlearning method trainers, benchmarks, evaluation metrics etc.) involves a similar procedure.

OU provides a modular framework where most of the Python implementation for dataset classes is shared across various dataset configurations and benchmarks. It also allows users to define custom dataset classes following the steps presented in Figure D.1. We already support three commonly used dataset handlers, each serving a distinct purpose in the unlearning pipeline:

- `PretrainingDataset`: used for training models on large-scale web corpora; essential for simulating pre-training settings.
- `CompletionDataset`: used for evaluating model outputs in a zero-shot or few-shot setting. This format is particularly useful for measuring memorization and information leakage, such as verbatim reproduction of forgotten content.
- `QADataset`: designed for probing models using natural language question-answer interactions, optionally with few-shot examples. This format is critical for assessing whether the model retains or forgets factual knowledge in interactive settings. Moreover, the framework automatically pipelines model-specific input formatting such as including system prompts or special tokens for chat-based models ensuring that queries are executed in a manner consistent with the model's native interface.
- `ForgetRetainDataset`: The unlearning process involves simultaneous optimization on both the forget and retain datasets, requiring concurrent batch loading. This dataset class abstracts this by loading the retain dataset in the same order as the forget dataset for unlearning.

Metrics OU supports multiple evaluation metrics and shares common functionalities across metric implementations. Metrics are broadly classified into three categories and summarized below:

Memorization Metrics: These metrics quantify how much the model has memorized information from its training data.

1. **Exact Memorization (EM):** Quantifies memorization by calculating proportion of tokens in the model's response that exactly match those in the ground truth y [Tirumala et al., 2022]. Formally, it is defined as

$$\text{EM} = \frac{1}{|y|} \sum_k \mathbf{1}\left\{\arg\max_y f(y \mid [x, y^{<k}]; \boldsymbol{\theta}) = y^k\right\}, \tag{D.1}$$

2. **Extraction Strength (ES):** Quantifies the intensity of memorization by determining the minimal prefix length required to reconstruct the remaining suffix [Carlini et al., 2021c].

$$\text{ES} = 1 - \frac{1}{|y|} \min_k \left\{k \mid f([x, y^{<k}]; \boldsymbol{\theta}) = y^{>k}\right\}. \tag{D.2}$$

3. **Probability (Prob.):** Directly quantifies the model's confidence in its output.

$$\text{Probability} = p\big(f(y \mid x)\big) \tag{D.3}$$

4. **Paraphrased Probability (Prob.):** Probability computed on a paraphrased answer y^{para} to remove template bias.

$$\text{Para. Prob.} = p\big(f(y^{\text{para}} \mid x)\big) \tag{D.4}$$

5. **ROUGE/Paraphrased ROUGE:** Assesses the degree of overlap between the model's output $f(x)$ and the ground truth y [Lin, 2004a]. This can be computed against many variants of datasets, including paraphrases and jailbreak prompts (next).
6. **Jailbreak ROUGE:** To probe for forgotten information, we employ a prefix-based jailbreaking attack by prompting the model with *"Sure, here is the answer:"* (as in [Wang et al., 2025a]) and then computing the ROUGE score between the model's response and the ground truth. This metric captures the extent to which suppressed content can still be recovered through prompt manipulation.
7. **Truth Ratio:** Measures the model's preference for the correct answer over a perturbed (incorrect) alternative by comparing their predicted probabilities. A higher value indicates stronger confidence in the correct response. It is defined as:

$$\text{Truth Ratio} = \frac{p(y^{\text{para}} \mid x)}{p(y^{\text{para}} \mid x) + p(y^{\text{pert}} \mid x)} \tag{D.5}$$

where y^{para} denotes the paraphrased correct answer and y^{pert} represents an incorrect alternative with similar structure. Note that Maini et al. [2024a] use a privacy-oriented variant of Truth Ratio computed as $\text{Truth Ratio} = min(\frac{p(y^{\text{para}}|x)}{p(y^{\text{pert}}|x)}, \frac{p(y^{\text{pert}}|x)}{p(y^{\text{para}}|x)})$.

We modify it so that it quantifies extent of knowledge for our work's purposes.

Privacy Metrics: These metrics ascertain whether sensitive information from the forget set can still be inferred or extracted from the model. Techniques such as Membership Inference Attacks (MIA) are utilized to evaluate the model's susceptibility to revealing whether specific data points were part of its training set, thereby assessing the privacy guarantees post-unlearning. However, these metrics often assume access to perfectly i.i.d. holdout splits or to an "oracle" retain model, limiting their practical usefulness in real-world settings.

1. **MIA:** Evaluates a model's tendency to memorize training data by testing whether an adversary can distinguish between seen examples from the forget set ($\mathcal{D}_{\text{forget}}$) and unseen examples from a holdout set ($\mathcal{D}_{\text{holdout}}$), based on model confidence. Ideally, a model that has not seen the forget set should yield an AUC of 0.5; however, due to challenges in constructing perfect holdout splits, benchmarks such as MUSE often calibrate this with AUC scores from the retain model (e.g., as done in PrivLeak). We support several MIA methods, including: LOSS [Yeom et al., 2018b], ZLib [Carlini et al., 2021c], GradNorm [Wang et al., 2024a], MinK [Shi et al., 2023b], and MinK++ [Zhang et al., 2025a].
2. **Forget Quality:** Performs a statistical test on the truth ratio distributions of the unlearned and retain models, yielding high values when the distributions closely match.

$$\text{KS}(\text{Truth Ratio}(f_{\text{target}}, \mathcal{D}_f), \text{Truth Ratio}(f_{\text{retain}}, \mathcal{D}_f)) \tag{D.6}$$

Utility Metrics: The goal of unlearning is to effectively forget the targeted data while preserving the model's performance on non-forget data. Utility metrics assess whether the model retains its capabilities on broader tasks beyond the retain data, ensuring that unlearning does not degrade general performance on real-world distributions.

1. **Model Utility (MU):** Captures the retained performance of a model after unlearning, both on the closely tied retain set and on broader general knowledge. TOFU computes MU as the harmonic mean of nine metrics across three data levels: the retain set, real authors, and factual world knowledge. At each level, it evaluates three metrics—probability, ROUGE, and the Truth Ratio.
2. **ROUGE for knowledge:** MUSE and TOFU assess utility by measuring ROUGE on knowledge-based questions.
3. **Forget Fluency:** Prior work [Mekala et al., 2025, Gandikota et al., 2024] has shown that unlearning often degrades model fluency, particularly on the forget set, resulting in random or nonsensical outputs. To capture this effect, we employ a classifier-based score that predicts whether a given text resembles gibberish[2].

[2] https://huggingface.co/madhurjindal/

4. **LM Eval Harness**: LM Evaluation Harness [Gao et al., 2024c] is an easy to use library enabling evaluations for a wide variety of general LLM benchmarks. It is integrated into OU, unlocking a broad suite of metrics such as WMDP MCQ, MMLU [Hendrycks et al., 2021b], GSM8K [Cobbe et al., 2021a] etc., for comprehensive post-unlearning evaluation.

By integrating the diverse metrics listed in Table 9.1, OU offers a robust framework to holistically evaluate unlearning methods, ensuring that models not only forget specific data but also maintain utility and privacy standards. Figure D.2 illustrates the process of adding a new metric to the OU framework.

It is important to recognize that the applicability of unlearning metrics often depends on the dataset used during evaluation. As a result, metrics implemented for one benchmark may not directly transfer to another. For example, the Knowledge Memorization metric in MUSE is based on question-answer pairs where answers are typically short, single-word responses. In contrast, TOFU lacks such a data split and instead features more descriptive, verbose answers. In this context, metrics like ROUGE recall may inadvertently capture surface-level template patterns rather than the core semantic content, potentially misleading the evaluation.

Models Different language models encode and store knowledge in fundamentally different ways depending on their architecture

(a) Metric Handler

```
@unlearning_metric(name =
"rouge")def rouge(model,* *
kwargs) : tokenizer =
kwargs["tokenizer"]data =
kwargs["data"]collator =
kwargs["collators"]batch_size =
kwargs["batch_size"]generation_args =
kwargs["generation_args"]...calculateROUGEreturn
```

(b) Metric Configuration

```
@package
eval.muse.metrics.forget_verbmem_ROUGEdefault
fillupforget_verbmem_ROUGE'sinputs'configs−
../../data/datasets@datasets :
MUSE_forget_verbmem −
../../collator@collators :
DataCollatorForSupervisedDatasetwithIndex−
../../generation@generation_args :
defaulthandler :
rougethehandlerwedefinedabovein(a)rouge_type :
rougeL_f1batch_size : 8datasets :
MUSE_forget_verbmem :
args : hf_args : path :
muse − bench/MUSE −
Bookspredict_with_generate :
Truecollators :
DataCollatorForSupervisedDataset :
args : padding_side :
leftforgenerationgeneration_args :
max_new_tokens : 128
```

Figure D.2: Example of a metric definition in OU: (a) the Python handler that implements the ROUGE metric, and (b) the corresponding configuration used to run ROUGE-based evaluation for assessing verbatim memorization.

and training setup. As a result, evaluating unlearning methods across a diverse range of models is essential for assessing their robustness and generalizability. However, existing benchmark implementations often support only a narrow set of model types and require users to manually rewrite evaluation logic such as input formatting, tokenization, and prompting—when adapting to new architectures. For example, chat-based models rely on specialized prompting structures that differ significantly from standard causal language models, making adaptation tedious and error-prone.

OU supports multiple model architectures and sizes out of the box. Built on Hugging Face Transformers [Wolf et al., 2020], it uses `AutoModelForCausalLM` and `AutoTokenizer`, while also supporting custom model loading (e.g., for probe models). A unified abstraction allows seamless switching between chat-style and base models without modifying the unlearning or evaluation pipeline, reducing overhead and enabling consistent cross-model comparisons.

In addition to support loading models in multiple precisions, OU also support loading 4-bit and 8-bit quantized models using the `bitsandbytes` library Dettmers et al. [2023]. This flexibility for quantization is particularly valuable for stress testing unlearning Zhang et al. [2025b].

New models for TOFU : OU provides trained models for the TOFU benchmark using LLaMA-based architectures finetuned on the TOFU dataset. These models span a range of sizes including 1B, 3B, and 8B parameters, enabling users to explore unlearning behavior across different model capacities. The 1B model, in particular, offers a highly efficient option for rapid experimentation with turnaround time of 15 minutes, requiring only 20 GB of GPU VRAM.

Model	Reference
LLaMA-2	Touvron et al. [2023]
LLaMA-3.1 / 3.2	Grattafiori et al. [2024]
Phi-1.5	Li et al. [2023f]
Phi-3.5	Abdin et al. [2024a]
Gemma	Gemma Team et al. [2024]
Zephyr	Tunstall et al. [2024]

Table D.1: Supported LLM Architectures in OU

Unlearning Methods Unlearning methods form the core of the OU framework. In practice, researchers proposing new unlearning approaches often evaluate them on a single benchmark due to the high efforts of adapting their code to other frameworks. This fragmentation has led to a lack of comprehensive, cross-benchmark comparisons in the unlearning literature. The overhead of re-implementing methods, adapting to different evaluation pipelines, and aligning metrics discourages reproducibility and slows progress.

OU addresses this gap by providing a unified and modular infrastructure that abstracts away benchmark-specific details. Researchers can implement their method once, typically by extending a custom `Trainer`,

and instantly evaluate it across multiple benchmarks. This design dramatically lowers the barrier to method development, evaluation and encourages the community to develop robust methods that work across benchmarks. We currently support all commonly used baselines as well as several state-of-the-art methods, and we invite the community to build upon this foundation.

Gradient Ascent [Maini et al., 2024a]: Performs gradient ascent on the forget set to degrade model confidence on targeted data.

$$= -\gamma \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \ell\left(y_f|x; f_{\text{unl}}\right) \tag{D.7}$$

GradDiff [Maini et al., 2024a]: Performs gradient ascent on forget data and descent on retain data.

$$= -\gamma \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \ell\left(y_f|x; f_{\text{unl}}\right) + \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

IdkNLL [Maini et al., 2024a]: Trains to output "I don't know" responses when queried on forgotten content.

$$= \gamma \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \ell\left(y_{\text{idk}}|x; f_{\text{unl}}\right) + \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

IdkDPO [Maini et al., 2024a]: Uses a DPO-style objective to align the model to output "I don't know" responses when queried on forgotten content.

$$= -\frac{2}{\beta} \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \log \sigma\Big(- \beta \log \left(\frac{p(y_{\text{idk}}|x; f_{\text{unl}})}{p(y_{\text{idk}}|x; f_{\text{target}})}\right) - \beta \log \left(\frac{p(y_f|x; f_{\text{unl}})}{p(y_f|x; f_{\text{target}})}\right)\Big)$$
$$+ \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

NPO [Zhang et al., 2024d]: Similar to the DPO-style objective, but uses only the negative feedback term in its formulation. It demonstrates better training stability compared to similar methods like GradDiff.

$$= -\frac{2}{\beta} \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \log \sigma\Big(- \beta \log \left(\frac{p(y_f|x; f_{\text{unl}})}{p(y_f|x; f_{\text{target}})}\right)\Big)$$
$$+ \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

SimNPO [Fan et al., 2024]: A modified variant of NPO that retains its core forgetting behavior **by** replacing the reference model with δ in the loss formulation.

$$= -\frac{2}{\beta} \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \log \sigma\Big(- \frac{\beta}{|y_f|} \log p(y_f|x; f_{\text{unl}}) - \delta\Big)\Big) + \alpha \mathbb{E}_{(}$$

AltPO [Mekala et al., 2025]: Uses a DPO-style objective to align the model toward generating alternate, in-domain plausible facts (produced by the model itself) that introduce ambiguity and suppress the original target knowledge.

$$= -\frac{2}{\beta} \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \log \sigma\Big(- \beta \log \left(\frac{p(y_{\text{alt}}|x; f_{\text{unl}})}{p(y_{\text{alt}}|x; f_{\text{target}})}\right) - \beta \log \Big($$
$$+ \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

RMU [Li et al., 2024b]: Assumes knowledge is encoded in model parameters and manipulates these representations to suppress memorization signals for the forget set while preserving knowledge in the retain set. Let $\phi(s; f_{\text{unl}})$ denote the embedding features

of the model, the loss is given by

$$= \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \frac{1}{|y_f|} \sum_{i=1}^{|y_f|} ||\phi([x, y^{<i}]; f_{\text{unl}}) - c \cdot \boldsymbol{u}||_2^2$$
$$+ \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \frac{1}{|y|} \sum_{i=1}^{|y|} ||\phi([x, y^{<i}]; f_{\text{unl}}) - \phi([x, y^{<i}]; f_{\text{target}})||_2^2$$

where $\boldsymbol{u}$ has elements randomly sampled from $[0, 1)$ and c is a scaling hyper-parameter.

UNDIAL [Dong et al., 2025]: Mitigates the instability found in prior methods by employing self-distillation, where the model learns from its own adjusted outputs. The core idea is to reduce the model's confidence in the target token by adjusting its logits, thereby diminishing its influence without affecting the overall model performance. This is achieved by minimizing the KL divergence between the adjusted logits and the model's current output distribution.

$$z_{\text{adj}}(x) = z_{\text{orig}}(x) - \beta \cdot \mathbf{1}_{y_f}$$
$$= \gamma \mathbb{E}_{(x,y_f)\sim\mathcal{D}_{\text{forget}}} \left[\text{KL}\left(\text{softmax}(z_{\text{adj}}(x)) \,||\, \text{softmax}(z_{\text{unl}}(x))\right)\right] + \alpha \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{retain}}} \ell\left(y|x; f_{\text{unl}}\right)$$

Where $z_{\text{orig}}(x)$ is the original logits produced by the model before unlearning and $z_{\text{adj}}(x)$ is the adjusted logits.

Technical improvements:

Efficiency: MUSE evaluates models without batching, while our implementation uses batched inference to improve efficiency. TOFU pads all sequences to a fixed `max_length` of 512, resulting in unnecessary GPU memory and compute overhead. In contrast, we apply dynamic padding based on the longest sequence in each batch. WMDP lacks a rigorous training and unlearning framework, limiting its extensibility for developing and evaluating new methods.

Training paradigms supported: Training or unlearning with larger models (e.g., ≥ 8B parameters) presents a significant computational challenge, often necessitating multiple high-end GPUs such as NVIDIA A100s. To accelerate this process, we support:

1. **DeepSpeed ZeRO Stage-3** [Jacobs et al., 2023]: Enabled via the Accelerate library [Gugger et al., 2022], reducing the memory usage through optimizer state partitioning and CPU/NVMe offloading.
2. **Model Parallelism**: Splits the model across GPUs along its layers, allowing large models to be trained even when individual GPUs cannot hold the full model in memory.

Experimental setup

All subsequent meta-evaluation and benchmarking experiments use the LLaMA-3.2-1B model. Experiments use BF16 precision, a single NVIDIA A100 GPU, a batch size of 32 and a paged AdamW optimizer (matching the TOFU paper's default settings).

Meta-evaluation

Faithfulness test-bed design We create two pools of models: the negative N and

the positive P pool. N contains models trained with varying training parameters while avoiding the knowledge of the forget set in the training data. P contains models trained similarly to N but with the target knowledge included in training. During the model pool preparation, we modify the training data used in the N and P pools with several training data variants. This introduces model diversity, forcing metrics to detect genuine *knowledge* retention rather than non-knowledge related artifacts, to achieve high scores. The faithfulness evaluation pipeline is illustrated in Figure 9.3 (a).

1. **Positive pool (P):** Models are trained on all TOFU facts (both `forget10` and `retain90`). We then replace `forget10` with two transformed variants. First, `forget10_paraphrased` uses paraphrased labels while preserving factual content. Second, `forget10_bio` contains long-form biographies derived from `forget10`.

2. **Negative pool (N):** Models are trained on the `retain90` split of TOFU, along with two perturbed variants of `forget10`. First, `forget10_perturbed` pairs each forget prompt with an incorrect label. Second, `celeb_bio` (biographies of random celebrities) serves as the counterpart to `forget10_bio`.

To further diversify the model pool, we vary training hyperparameters: five learning rates from 1×10^{-5} to 5×10^{-5}, and two checkpoints (after training epochs 5 and 10). Combining 2 pools $\times$ 3 dataset variants $\times$ 5 learning rates $\times$ 2 checkpoints yields 60 models in total.

Data generation process While some of TOFU's evaluation datasets include paraphrased and perturbed examples, our training-set variants for the model pool were generated independently. We used LLaMA 3.1 405B via the SambaNova API[3] to paraphrase and perturb QA pairs, and prompted Gemini[4] to produce Wikipedia-style biographies from each author's 20 QA pairs.

Robustness setup design We create a large and diverse pool of unlearned models and a separate set of retain models, which serve as gold-standard references having never been trained on the forget set. The unlearned pool is then subjected to stress-test interventions, to provoke recovery (or inducing) of the forgotten knowledge. These pools serve as our test-bed. For every metric being meta-evaluated, values are recorded on both pools before and after each intervention. The change in a metric's distribution before and after intervention on the unlearned models (along with the change in retain models for normalization) is used to characterize robustness. We use three interventions: *relearning*, *quantization* and *probing*.

1. **Relearning Setup:** We finetune the unlearned model on the full `forget10` dataset for one epoch with a learning rate of 2×10^{-5}.

[3] https://cloud.sambanova.ai/playground
[4] `gemini-2.0-flash-exp` (accessed 26 April 2025)

2. **Quantization Setup:** We apply 4-bit floating-point quantization using BitsAndBytes [Dettmers et al., 2023]. Checkpoints unlearned with a learning rate of 1×10^{-5} are chosen, as quantization is most effective at lower learning rates [Zhang et al., 2025b].

3. **Probing:** We evaluate layer 11 of the LLaMA-3.2-1B model (16 layers total) using the language-model head from the corresponding `retain90`-trained model. This head is trained with a learning rate of 1×10^{-4} on `retain90` for ten epochs.

Additional Results Figure D.4 shows the faithfulness of the metrics, while Figure D.5 and Figure D.6 show their behavior under relearning and quantization stress tests. We found that removing MU filter of retaining at least 80% utility for unlearned models reduces robustness to quantization further (see Figure D.7). Despite this, we apply the MU filter to better align with common unlearning reporting practices.

Probing results: We compute the metric robustness to probing intervention as follows

$$p = \frac{m^a_{\text{ret}}}{m^a_{\text{unl}}} \quad \text{if} \quad \frac{m^b_{\text{ret}}}{m^b_{\text{unl}}} \geq 1, \quad P = \min(p, 1) \tag{D.8}$$

Table D.2 shows the results of our metric meta-evaluation with probing. Probing, while provided for by OU, is not used in the meta-evaluation procedure, as P scores on TOFU achieve 1 for all metrics and thus offer little information.

Metrics	Probe ↑
Exact Mem.	1.0
Extr. Strength	1.0
Truth Ratio	1.0
Prob.	0.99
ROUGE	0.99
Jailbreak ROUGE	0.99
Para. Prob.	1.0
Para. ROUGE	0.99
MIA - LOSS	1.0
MIA - MinK	1.0
MIA - MinK++	0.83
MIA - ZLib	1.0

Table D.2: Robustness meta-evaluation with probing (layer 11)

Further considerations

Why aren't the intervened versions of metrics considered evaluation metrics themselves? The interventions we use require modification to and access of model weights, which an unlearning auditor might not possess. In the case of relearning and quantization, they also involve computational costs associated with training and calibration. Stress-testing interventions are best suited for final-stage audits before model deployment, rather than for routine use throughout unlearning workflows, as is expected of standard evaluation metrics. Our analysis can inform the design of robust evaluation metrics that function without requiring stress-testing.

Comparison to Wang et al. [2025a]'s meta-evaluation: Our work is related to the recent effort by Wang et al. [2025a] to com-

pare unlearning evaluation metrics. Their analysis focuses on four metrics: probability, ROUGE, ES, and EM, and evaluates robustness by measuring the linear correlation of metric values before and after applying stress-tests such as jailbreaking, relearning, probing, and token noising. We extend this framework in several key ways.

1. **Broader metric coverage:** We evaluate a broader range of metrics, including six additional ones.
2. **Faithfulness assessment:** We assess faithfulness of metrics in our meta-evaluation as a minimal criterion. This enforces that good metrics must accurately capture the presence or absence of target knowledge, rather than merely resisting change under intervention.
3. **Focused interventions:** We focus specifically on three interventions: relearning, probing, and quantization, excluding jailbreaking and token-noising from the intervention set. We instead treat jailbreaking as an evaluation metric in its own right. Prompt-based attacks like paraphrasing and jailbreak-style prompts are more naturally seen as inexpensive evaluation metrics rather than stress-testing interventions. Additionally, Wang et al. [2025a] found jailbreaking and token noising (which is also a prompt modification) to be less effective as interventions.
4. **A different calibration criterion:** Our procedure also introduces a calibration criterion grounded in ideal behavior. Rather than expecting linear variation from a metric upon intervention, we benchmark metric behavior against a gold-standard retain model, for a more principled signal of robustness.
5. **Practical robustness analysis:** Our robustness analysis filters for models with good utility that are substantially unlearned, selected from a diverse and representative set of unlearning algorithms. This leads to a test distribution for metrics that better reflects realistic unlearning scenarios.

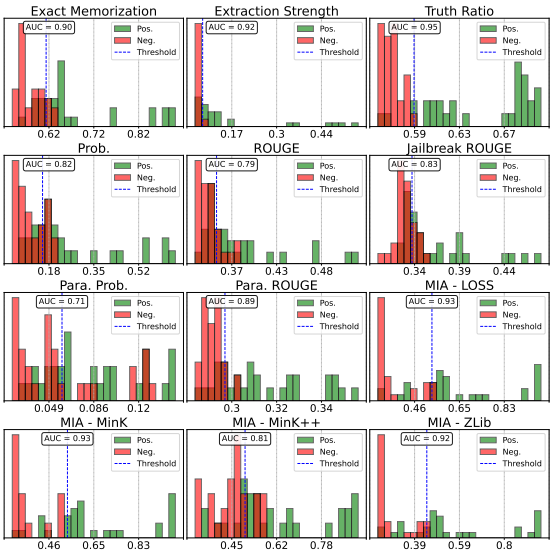


Figure D.4: Faithfulness: Evaluation of multiple metrics to assess faithfulness. AUC indicates how effectively metrics distinguish between models trained on the target knowledge and those that are not.

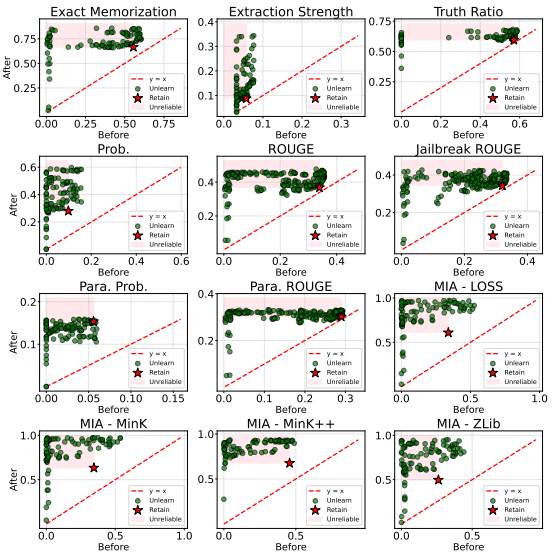


Figure D.5: Relearning: Stress-testing multiple evaluation metrics through relearning. A significant fraction of unlearned models regain knowledge faster than the retained model when re-exposed to the forgotten data, falling into the unreliable red-shaded region: indicating that the metrics failed to initially capture the knowledge and are thus not robust.

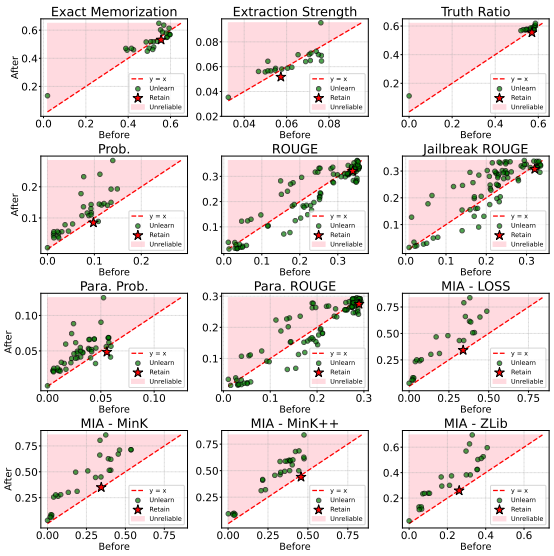


Figure D.6: Quantization: Stress-testing multiple evaluation metrics through quantization. For several metrics, a subset of unlearned models shows increased metric values after quantization, falling into the red-shaded region: suggesting that the metrics failed to initially capture the presence of knowledge and are therefore not robust. These results are reported only for models unlearned with low learning rates and high utility.

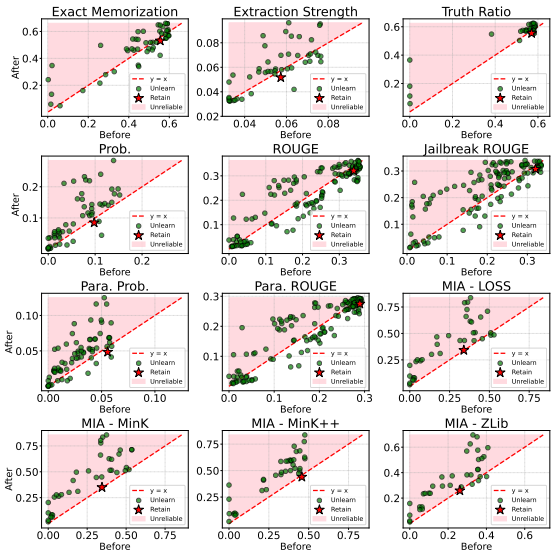


Figure D.7: Quantization: Stress-testing multiple evaluation metrics through quantization. For each metric, a subset of unlearned models shows increased metric values after quantization, falling into the red-shaded region, suggesting that the metrics failed to initially capture the presence of knowledge and are therefore not robust. These results are reported only for models unlearned with low learning rates and no filter on utility.

Further discussion on benchmarking unlearning methods

Metric aggregation There are theee dimensions evaluated by our suite of metrics 1) Memorization, 2) Privacy 3) Utility.

We consider multiple metrics in each dimension and aggregate the score as follows:

1. **Memorization**: To quantify the degree of successful forgetting, the Memorization Score is calculated as the Harmonic Mean (HM) of 4 core metrics which are best as per our meta-evaluations analysis in Sec. 9.2 — ES, EM, Paraphrased Probability and Truth Ratio. These metrics are inverted (i.e., 1 − metric) so that higher scores indicate more effective unlearning. The score is given by:

   $$\text{Memorization Score} = \text{HM}\,(1 - \text{ES},\ 1 - \text{EM},\ 1 - \text{Para.}$$

2. **Privacy**: For assessing privacy, we utilize four Membership Inference Attack (MIA) metrics: LOSS, ZLib, Min-k, and Mink++. For each of these, an individual privacy score (s_{MIA}) is calculated. This score, ranging from 0 to 1, quantifies how closely the unlearned model's behavior on the specific MIA metric aligns with that of a gold-standard retain model (details below). A higher s_{MIA} score indicates greater similarity to the retain model. The overall Privacy Score is then the Harmonic Mean (HM) of these individual scores:

   $$\text{Privacy Score} = \text{HM}(s_{\text{LOSS}}, s_{\text{ZLib}}, s_{\text{Min-k}}, s_{\text{Mink++}})$$

3. **Utility**: TOFU evaluates a model's utility using nine core metrics that assess performance across splits at three different distances from the forget dataset distribution - namely, retain, real-world authors, and wrong-fact queries: using QA probability, ROUGE, and truth-ratio scores. In addition to this we include a new metric that measures the fluency of the model's response when prompted with entities-related to forget queries, following [Mekala et al., 2025, Gandikota et al., 2024]. Fluency is assessed using a classifier[5] that detects gibberish / nonsensical outputs. The final utility score is the harmonic mean of MU and fluency. Note that we scale all metrics with init finetuned model, so their scores across all points fall in the $[0, 1]$ range. For example, TOFU MU scores never exceed that of the initial target model upon unlearning, so all scores are effectively divided by the target model's MU.

Note that for many metric aggregations we use Harmonic Mean, as HM ensures that a high final score demands strong performance in all constituent parts.

Hyperparameter tuning and model selection while comparing unlearning methods

Hyperparameters used

1. For GradDiff and IdK-NLL: we vary the learning rate over the set $\{1 \times 10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}, 4 \times 10^{-5}, 5 \times 10^{-5}\}$, and sweep the regularization coefficient $\alpha \in \{1, 2, 5, 10\}$.
2. For IdK-DPO, NPO and AltPO: we tune learning rates in $\{1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$, and search over $\alpha \in \{1, 2, 5\}$ and $\beta \in \{0.05, 0.1, 0.5\}$.
3. For RMU: we use the same learning rate range $\{1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$, vary the steering coefficient in $\{1, 10, 100\}$, and apply the loss at one of the layers $l \in \{6, 11, 16\}$ of the LLama3.2-1B model. For each selected layer l, we restrict training to layers $l - 2$, $l - 1$, and l.
4. For SimNPO: we tune learning rates in $\{1 \times 10^{-5}, 2 \times 10^{-5}, 5 \times 10^{-5}\}$, and search over $\beta \in \{3.5, 4.5\}$, $\delta \in \{0, 1\}$ and $\delta \in \{0.125, 0.25\}$.
5. For UNDIAL: we tune learning rates in $\{1 \times 10^{-5}, 1 \times 10^{-4}, 3 \times 10^{-4}\}$, and search over $\alpha \in \{1, 2, 5\}$ and $\beta \in \{3, 10, 30\}$.

We aggregate utility score and memorization score and use their harmonic mean for tuning the models.

What metrics are appropriate for model selection during hyperparameter tuning? The nature of tuning in unlearning benchmarking has distinct considerations compared to general machine learning. While standard machine learning avoids using test data for tuning to ensure generalization, unlearning in TOFU and MUSE specifically targets the known forget set for erasure. Consequently, iteratively refining the unlearning by evaluating the model's behavior concerning this specific set is a permissible ap-

[5] https://huggingface.co/madhurjindal/autonlp-Gibberish-Detector-492513457

proach to ensure thorough forgetting before deployment. For this tuning, we advocate relying on metrics realistically available during the development phase, specifically those assessing forget quality on the target data and general utility, while avoiding "oracle" metrics that presume access to unavailable resources like true i.i.d holdout sets or retain models like in [Maini et al., 2024a, Shi et al., 2025]. Since all our privacy scores use a retain model, we avoid them during tuning. We rely on the harmonic mean of the Memorization and Utility scores as the validation objective.

Comparison to Wang et al. [2025a]'s benchmarking: While Wang et al. [2025a] propose approaches towards model selection and benchmarking through validation on Extraction Strength and calibration via model-merging, their analysis has several limitations. They rely only on ES scores for evaluating forgetting and utility. ES was found to be robust among the set of 4 evaluation metrics (an observation also re-verified in our work (Sec. 9.4). Yet it has not been proved that ES is a comprehensive metric validating all facets of knowledge unlearning. For example, ES does not account for privacy metrics that prevent over-unlearning, like TOFU's Truth Ratio or FQ or MUSE's PrivLeak. In addition, they do not consider all facets of general utility evaluation, particularly forget set fluency. Finally, the question of what metrics can be used in model selection and if they must be separate from the leaderboard metrics remains unanswered. These limitations remain, to a smaller degree, in our benchmarking procedure, and we consider this an important line for further research.

Method	**Agg. ↑**	**Mem. ↑**	**Priv.**
Init. finetuned	0.00	0.00	0.10
Retain	0.58	0.31	1.00
GradDiff [Maini et al., 2024a]	**0.87**	**0.97**	3.27e-
AltPO [Mekala et al., 2025]	<u>0.76</u>	<u>0.63</u>	0.06
IdkDPO [Maini et al., 2024a]	0.71	0.56	0.06
NPO [Zhang et al., 2024d]	0.69	0.52	0.06
RMU [Li et al., 2024b]	0.53	0.47	0.5
SimNPO [Fan et al., 2024]	0.49	0.32	<u>0.63</u>
UNDIAL [Dong et al., 2025]	0.4	0.27	0.48
IdkNLL [Maini et al., 2024a]	0.14	0.08	0.17

Table D.3: Comparison of unlearning methods on the TOFU task, showing aggregate (Agg.) using only Memorization (Mem.) and utility (Utility) scores. Privacy scores are not used in the aggregation and are only shown for illustration. Higher scores indicate better performance (↑). Initial finetuned is the target model before unlearning and Retain model is the gold standard target model. The focus on memorization as opposed to privacy results in GradDiff performing the best as it easily results in over-unlearning.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer:

   Justification: Yes, our paper's main contributions are precisely summarized in the introduction and also at a higher level in the abstract.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer:

   Justification: We include the limitations of our work in Appendix 16.6.2.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often

depend on implicit assumptions, which should be articulated.

- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer:

   Justification: This is not a theoretical paper.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.

- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer:

   Justification: We provide the experimental configuration in Appendix D; meta-evaluation procedures in Sec. 9.4 and Appendix D; hyperparameters for unlearning in Appendix D; and metric aggregation details in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture,

the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer:

   Justification: GitHub links for code, and HuggingFace links for models are provided in the footnotes of Sec. D.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionP for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset

of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer:

   Justification: We discuss data-splits in Appendix D, the training details in Sec. 9.5 and Appendix Sec. D. We provide the experimental configuration in Appendix D; meta-evaluation procedures in Sec. 9.4 and Appendix D; hyperparameters for unlearning in Appendix D; and metric aggregation details in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer:

   Justification: NA

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer:

   Justification: We mention the details in Appendix D.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer:

   Justification: We have read the Code.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer:

    Justification: Discussed in Appendix 16.6.2.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer:

    Justification: We release finetuned LLaMA models trained on TOFU, a dataset of fictitious biographies, which poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer:

    Justification: Yes, we cite their work and use their implementations in accordance with licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are repackaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer:

    Justification: Yes, this paper and in addition the documentation in our GitHub repository, thoroughly documents the available features.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/-model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourc-

ing nor research with human subjects.

- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer:

    Justification: NA

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer:

    Justification: NA

    Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (https://neurips.cc/Conference 2025/LLM) for what should or should not be described.

(a) LLaMA 3.2 1B model configuration

```
model_args                                    :
pretrained_model_name_or_path                 :
"meta-llama/Llama-3.2-1B-Instruct"attn_implementation :'
flash_attention'_2torch_dtype                 :
bfloat16tokenizer_args                        :
pretrained_model_name_or_path                 :
"meta-llama/Llama-3.2-1B-Instruct"template_args :
apply_chat_template                           :
Truesystem_prompt                             :
Youareahelpfulassistant.date_string :
10Apr2025
```

(b) LLaMA 2-7B model configuration

```
model_args                                    :
pretrained_model_name_or_path                 :
"meta-llama/Llama-2-7b-hf"attn_implementation :'
flash_attention'_2torch_dtype                 :
bfloat16tokenizer_args                        :
pretrained_model_name_or_path                 :
"meta-llama/Llama-2-7b-hf"template_args       :
apply_chat_template                           :
Falseuser_start_tag : "Question               :
"user_end_tag : ""asst_start_tag              :
"Answer :"asst_end_tag :""
```

Figure D.3: Example model configurations for two different LLaMA variants: (a) LLaMA 3.2-1B with chat template prompting, and (b) LLaMA 2-7B with manual prompt formatting.

Chapter E

SSFT

Characterizing Datapoints via Second-Split Forgetting Supplementary Material

Theoretical Results

Preliminaries Let $\mathbf{w} \in \mathbb{R}^d$ represent the weight vector of overparametrized linear model. We analyze the learning and forgetting dynamics by minimizing the empirical risk: $\mathcal{L}(\mathcal{D}; \mathbf{w}) = \sum_i \ell(\mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i)$. We consider the exponential loss $\ell(z) = \exp(-z)$ for our analysis. For completeness, we rewrite the definitions and preliminaries from the main paper below.

Data Generating Process We restate the data generating process as detailed in the synthetic experiment in Section 10.4.2. We consider data $(\mathbf{x}, \mathbf{y})$ sampled from a mixture of multiple distributions $\mathcal{D}_g$, s.t. $\mathbf{x} \in \mathbb{R}^d$. $\mathcal{D}_g$ denotes the g^{th} group and has a sampling frequency of π_g. Each group $\mathcal{D}_g$ is a distribution over $(\mathcal{D}_{Xg} \times \{\mathbf{y}_g\})$, i.e., the true label for all the samples drawn from a given group is the same, and the examples in each group are non-overlapping. Each group is parametrized by a set of $kL^2(\Omega)d$ unique indices $\mathcal{I}_g \subset [d]$ such that $\mathcal{I}_i \cap \mathcal{I}_j = \phi$ for $i \neq j$. The discriminative characteristic of each group is the vector $\boldsymbol{u}_g$, such that, $[\boldsymbol{u}_g]_i = 1$ if $i \in \mathcal{I}_g$ else 0 $\forall i \in [d]$. In the following discussion, we will refer to μ_g as the coordinate-wise mean vector for the group g, such that $\boldsymbol{\mu}_g = \mu_g \boldsymbol{u}_g$. Then for any sample $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}$:

$$P(\mathbf{x} \in \mathcal{D}_{Xg}) = \pi_g; \quad \mathbf{x}|\mathcal{D}_{Xg} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_d) + \boldsymbol{\mu}_g. \tag{E.1}$$

Definition 1 (Rare Examples, $\mathcal{A}$ [Feldman, 2020a]). *Consider a dataset $\mathcal{D}$ sampled from a mixture of distributions $\{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ with frequency $\{\pi_1, \ldots, \pi_N\}$ respectively. Let $\mathcal{A} \subseteq \mathcal{D}$ be the set of rare examples. Then, for all $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{A}$, if $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_j$, then there are $O(1)$ samples from $\mathcal{D}_j$ in $\mathcal{D}$.*

Definition 2 (Mislabeled Examples, $\mathcal{M}$). *Consider a k class classification problem with $\mathcal{D}_Y = \{1, 2, \ldots, k\}$. Let $\mathcal{M} \subset \mathcal{D}$ be the set of mislabeled examples. Then for any $(\mathbf{x}, \mathbf{y}) \sim \mathcal{D}$, a corresponding mislabeled example is given by $(\mathbf{x}, \tilde{\mathbf{y}}) \in \mathcal{M}$ such that $\tilde{\mathbf{y}} \in \mathcal{D}_Y \setminus \{\mathbf{y}\}$.*[1]

Definition 3 (Complex Examples, C). *Let $C \subset \mathcal{D}$ be the set of examples sampled from complex distributions. Let $(\mathbf{x}_i, \mathbf{y}_i) \in C$ such that $(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_g$ (complex group), then $\mu_g = \frac{\mu_t}{\lambda}$, $\lambda > 1$ where μ_t is the coordinate-wise mean for samples drawn from any simple distribution $\mathcal{D}_t$ (Section 10.4.2).*

The implication of the aforementioned characterization is that complex distributions have a lower signal-to-noise ratio (SNR)

[1] For binary classification, $\mathcal{D}_Y = \{-1, +1\}$. The labels are reversed for mislabeled examples.

as compared to simple distributions. We assume the sample complexity required to estimate the distribution as a proxy for the complexity of the distribution. In this regard, having a low SNR increases the complexity.

Recall that $\mathcal{D}_A$ and $\mathcal{D}_B$ denote the first and second training splits of our dataset. In our theoretical framework, we only consider two majority distributions in the second split dataset ($\mathcal{D}_B$). Let us call them $\mathcal{D}_1, \mathcal{D}_2$. Therefore, both $\mathcal{D}_A$ and $\mathcal{D}_B$ contain $O(n)$ samples from $\mathcal{D}_1, \mathcal{D}_2$. In the first split dataset ($\mathcal{D}_A$), we consider the presence of another distribution $\mathcal{D}_r$ that constitutes the rare example (only one sample $(\mathbf{x}_r, \mathbf{y}_r)$). The mislabeled example (only one sample $(\mathbf{x}_m, \mathbf{y}_m)$) belongs to one of the majority distributions, and we will assume without loss of generality that this is from distribution $\mathcal{D}_1$. To understand the population accuracy in the case of complex distributions, we will draw a simple analogy in the case when the majority distributions $\mathcal{D}_1, \mathcal{D}_2$ occur from complex distributions as defined below. Based on equation E.1, without loss of generality, we will assume that dimensions $\{1 \dots k\}, \{k+1 \dots 2k\}, \{2k+1 \dots 3k\}$ are the predictive dimensions for the majority group 1, 2 and that for the rare example from dataset 1. We make these assumptions to simplify the theoretical exposition. However, our results can be observed even after relaxing them at the expense of more book-keeping.

Based on Chatterji and Long [2021], we make the following assumptions about the problem setup:

(A.1) The failure probability satisfies $0 \leq \delta \leq 1/C$,

(A.2) The number of samples satisfies $n \geq C \log(1/\delta)$,

(A.3) The input dimension $d \geq C \max\{n^2 \log(n/\delta), nk\mu$ and $k \cdot \mu^2/\sigma^2 \geq C \log(n/\delta)$,

where $k \cdot \mu^2/\sigma^2$ represents the signal to noise ratio.

Optimization We perform gradient descent with fixed learning rate η,

$$\mathbf{w}^{(t+1)} = \mathbf{w}(t) - \eta \nabla \mathcal{L}(\mathbf{w}(t)) = \mathbf{w}(t) - \eta \sum_i \ell'(\mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i) \cdot \mathbf{y}_i \mathbf{x}_i. \tag{E.2}$$

Asymptotic Solution [Soudry et al., 2018] For sufficiently small learning rate η, and (bounded) starting point $\mathbf{w}(0)$,

$$\mathbf{w}(t) = \hat{\mathbf{w}} \log t + \rho(t), \tag{E.3}$$

where $\hat{\mathbf{w}}$ is the solution to the hard margin SVM:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{R}^d} \|\mathbf{w}\|_2^2 \quad s.t. \ \mathbf{w}^\top \mathbf{y}_i \mathbf{x}_i \geq 1, \tag{E.4}$$

Learning Stage For the stage 1, we consider that we train the model for a maximum of T epochs (until we achieve 100% accuracy on the first training dataset $\mathcal{D}_A$ with margin greater than 1). This means that the learned weight vectors are close to, but have not converged to the max margin solution. The solution at the end of T epochs is given by

$\mathbf{w}_A(t)$. At sufficiently large T, we have:

$$\begin{aligned} \mathbf{w}_A(T) &= \hat{\mathbf{w}}_A \log T + \rho_A(T) \\ \mathbf{w}_A(T)^\top \mathbf{y}_i \mathbf{x}_i &\geq 1 \quad \forall (\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_A \end{aligned} \tag{E.5}$$

Lemma 1 (Bounded Weights)**.** *With probability at least $1-\delta$, there exists a bounded time t beyond which the model classifies all training points correctly.*

Proof. From Lemma 5, we know that the dataset is separable with probability at least $1-\delta$. This means that the max-margin solution (or the SVM solution) for the dataset classifies all training points correctly. From the analysis in Soudry et al. [2018], we know that:

$$\begin{aligned} \lim_{t\to\infty} \frac{\mathbf{w}_A(t)}{\|\mathbf{w}_A(t)\|_2} &= \frac{\hat{\mathbf{w}}_A}{\|\hat{\mathbf{w}}_A\|_2}, \\ \lim_{t\to\infty} \mathbf{y}_i \mathbf{x}_i^\top \frac{\mathbf{w}_A(t)}{\|\mathbf{w}_A(t)\|_2} &= \mathbf{y}_i \mathbf{x}_i^\top \frac{\hat{\mathbf{w}}_A}{\|\hat{\mathbf{w}}_A\|_2} \end{aligned} \tag{E.6}$$

Then it directly follows that, for $\epsilon > 0$, there exists bounded time $T > 0$ such that $\forall t > T$,

$$\text{abs}\mathbf{y}_i \mathbf{x}_i^\top \frac{\mathbf{w}_A(t)}{\|\mathbf{w}_A(t)\|_2} - \mathbf{y}_i \mathbf{x}_i^\top \frac{\hat{\mathbf{w}}_A}{\|\hat{\mathbf{w}}_A\|_2} < \epsilon$$

This concludes that there exists a time T for which the sign of the prediction of both the max-margin solution and the learnt solution will be the same, implying correct prediction for every example in the training set for a bounded time solution.

□

Forgetting Stage We initialize the weights for second stage of training with $\mathbf{w}_A(T)$ from first training stage, and then train on $\mathcal{D}_B$ to minimize the empirical loss using gradient descent (Equation 10.1). Assume that the dataset is balanced in the class labels, $|\mathcal{D}_B| = n$. Also, recall that $\mathcal{D}_B$ does not contain mislabeled or rare examples from the same subgroup as in $\mathcal{D}_A$. For analyzing example forgetting in an asymptotic setting, we will directly borrow results from the analysis made by Chatterji and Long [2021]. They prove a stronger result for the case where the dataset contains a fraction η of mislabeled examples. However, we will use the setting without label noise. The asymptotic result then builds on to the main theorem of the paper on intermediate time forgetting (Theorem 6).

Transformations for Equivalence to Chatterji and Long [2021] In our setup, we consider a group structure where each distribution has a mean vector that is orthogonal to the others. We show the equivalence of the same to the data model studied in prior work [Chatterji and Long, 2021].

Let us define $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \in \mathbb{R}^d$ as follows:

$$\boldsymbol{\mu}_1 j = \begin{cases} \mu & \text{if } j \in \{1 \dots k\} \\ 0 & \text{o.w.} \end{cases}$$

Similarly, define $\boldsymbol{\mu}_2, \boldsymbol{\mu}_3$ as well. For the equivalence condition, we are only concerned about the dataset split $\mathcal{D}_B$ which is where the generalization bounds hold. Further, let $z \sim \mathcal{N}\left(0, \sigma^2 \mathbf{I}_d\right)$. Then, $\mathbf{x}|\mathcal{D}_{Xg} \sim z + \boldsymbol{\mu}_g$.

We can now shift and rescale the axes such that the new origin is located at $(\mu_1+\mu_2)/2$.

Then, define $\boldsymbol{\mu} = (\mu_1 - \mu_2)/2$. This results in the simplification that the mean of the examples sampled from $\mathcal{D}_1$ is $\boldsymbol{\mu}$ and that from $\mathcal{D}_2$ is $-\boldsymbol{\mu}$. We can hence express $\mathbf{x} = \mathbf{y}\boldsymbol{\mu} + \mathbf{z}$. This directly follows their model assumption where $\mathbf{z} \in \mathbb{R}^d$ has each marginal sampled from a mean zero subgaussian distribution with subgaussian norm at most 1. In our case, each marginal is a Gaussian random variable with variance σ^2. Once again, we can rescale the axes such that $\tilde{\boldsymbol{\mu}} = \mu/\sigma$. Now, our data model directly follows the data model discussed in prior work [Chatterji and Long, 2021].

Asymptotic Analysis First, we analyze infinite-time training case. We will extend the result from Soudry et al. [2018], Chatterji and Long [2021] to show that (i) mislabeled and rare examples are forgotten when the model is trained for long; and (ii) complex examples are not forgotten. First, recall the result used in Subsection E for any bounded initialization for weights $\mathbf{w}_B(0)$,

$$\mathbf{w}_B(t) = \hat{\mathbf{w}}_B \log t + \rho_B(t). \tag{E.7}$$

We will first provide a formal version of Theorem 3 which was informally stated in the main paper.

Theorem 5 (Asymptotic Forgetting)**.** *Under assumptions A.1, A.2, A.3, with probability 1-δ, fine-tuning for $t \to \infty$ iterations on the second dataset $\mathcal{D}_B$ produces a max-margin classifier $\hat{\mathbf{w}}_B$ such that*

$$\begin{aligned}
&\qquad\qquad\qquad\qquad (\mathbf{x}_m, \mathbf{y}_m) \in \mathcal{D}_A \left[\operatorname{sign}(\hat{\mathbf{w}}_B \cdot \mathbf{x}_m) = \mathbf{y}_m\right] \le \exp\left(-c\|\tilde{\mu}\|_2^2/\right.\\
\Phi\left(-1/C\right) \le\ &{}_{(\mathbf{x}_r,\, \mathbf{y}_r) \in \mathcal{D}_A} \left[\operatorname{sign}(\hat{\mathbf{w}}_B \cdot\ \mathbf{x}_r) \neq\ \mathbf{y}_r\right] \le \Phi\left(1/C\right),\\
&{}_{(\mathbf{x}_c,\, \mathbf{y}_c) \in \mathcal{D}_A} \left[\operatorname{sign}(\hat{\mathbf{w}}_B \cdot\ \mathbf{x}_c) \neq\ \mathbf{y}_c\right] \le \exp\left(-c\|\tilde{\mu}/\lambda\|\right.
\end{aligned}$$

for some absolute constant c>0 and $(\mathbf{x}_m, \mathbf{y}_m) \in \mathcal{M}, (\mathbf{x}_r, \mathbf{y}_r) \in \mathcal{A}, (\mathbf{x}_c, \mathbf{y}_c) \in C$ respectively.

The theorem implies that the probability that mislabeled examples from $\mathcal{D}_A$ are classified with the given (incorrect) label tends to 0 if $\|\boldsymbol{\mu}\|_2 = \theta(d^\beta)$ for any $\beta \in (1/4, 1/2]$. Note that in our case, we have k dimensions of signal. Therefore, as long as k/d is a constant fraction, as the input dimensions $d \to \infty$ the above holds. Examples in $\mathcal{D}_A$ from distributions absent in $\mathcal{D}_B$ are randomly classified.

Intuition. In the asymptotic case, the initial weights from first split training $\mathbf{w}_A(T) = O(\log T) L^2(\Omega) \mathbf{w}_B(t)$, where $t \to \infty$ in the limit of infinite training. As a consequence, for any bounded initialization, the model weights converge to the minimum norm solution (SVM) solution from Soudry et al. [2018]. We use results from [Chatterji and Long, 2021] who study the setting of a binary classification problem with noisy label fraction η. In our case, since the second-split is assumed to have only clean samples, $\eta = 0$. The final step is to adapt our data generating process to the format used in Chatterji and Long [2021].

Proof. Recall the transformations described in Section E. Let $(\mathbf{x}_m, \mathbf{y}_m), (\mathbf{x}_r, \mathbf{y}_r), (\mathbf{x}_c, \mathbf{y}_c)$ represent any point from $\mathcal{D}_A$ which belongs to mislabeled set $\mathcal{M}$, rare set $\mathcal{A}$ and complex

set C. The important thing to note in the analysis that follows is that each of these examples is independent of the samples in $\mathcal{D}_B$. Hence, the probability of correctly predicting on them is same as that of correctly predicting on a population sample, in the limit of infinite training (when initialization does not matter and all models converge to the same solution).

$$_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[\text{sign}(\mathbf{w}\cdot\mathbf{x})\neq\mathbf{y}\right] =_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[(\mathbf{y}\mathbf{w}\cdot\mathbf{x})<0\right]$$

We will now analyze the probability of correct prediction of mislabeled, rare, and complex examples separately.

Mislabeled Examples Since $\mathcal{D}_B$ is separable, in the limit of infinite-training time, the classifier correctly predicts all the examples in the dataset. We will use this fact to show that it assigns the opposite label to the mislabeled sample in set $\mathcal{D}_A$ with high probability. Then, we denote 'failure' as the event that the mislabeled samples is still predicted with label $\mathbf{y}_m$ at infinite-time training.

$$\begin{aligned}
_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_1}\left[(\mathbf{y}\mathbf{w}\cdot\mathbf{x})<0\right] &= 1-\left[(\mathbf{y}_m\mathbf{w}\cdot\mathbf{x}_m)<0\right] \\
&= \left[(\mathbf{y}_m\mathbf{w}\cdot\mathbf{x}_m)>0\right] \\
&= \left[\text{sign}(\mathbf{w}\cdot\mathbf{x}_m)=\mathbf{y}_m\right],
\end{aligned}$$

Then, we can directly borrow the result from Chatterji and Long [2021] (Theorem 4) to find that

$$\begin{aligned}
_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_1}\left[(\mathbf{y}\mathbf{w}\cdot\mathbf{x})>0\right] &\leq \exp\left(-c\frac{||\tilde{\boldsymbol{\mu}}||^4}{d}\right) \\
\left[\text{sign}(\mathbf{w}\cdot\mathbf{x}_m)=\mathbf{y}_m\right] &\leq \exp\left(-c\frac{||\tilde{\boldsymbol{\mu}}||^4}{d}\right),
\end{aligned} \tag{E.8}$$

Rare Examples Without loss of generality, we may assume that the correct label for the rare example $\mathbf{y}_r = 1$.

$$\begin{aligned}
[(\mathbf{y}_r\mathbf{w}\cdot\mathbf{x}_r)<0] &= \left[(\mathbf{y}_r\mathbf{w}\cdot(\mathbf{x}_r-\boldsymbol{\mu}_r)<-\mathbf{y}_r\mathbf{w}\cdot\boldsymbol{\mu}_r\right] \\
&= \left[(\mathbf{w}\cdot(\mathbf{x}_r-\boldsymbol{\mu}_r)<-\mathbf{w}\cdot\boldsymbol{\mu}_r\right], \qquad \text{(since } \mathbf{y}_r \\
&= \left[(\mathbf{w}\cdot\mathbf{z}_r<-\mathbf{w}\cdot\boldsymbol{\mu}_r\right] \\
&= \Phi\left(\frac{-\mathbf{w}\cdot\boldsymbol{\mu}_r}{\sigma\,\|\mathbf{w}\|_2}\right),
\end{aligned} \tag{E.9}$$

where Φ is the Gaussian CDF. In the last step we use the fact that $\mathbf{z}_r \sim \mathcal{N}(0,\sigma^2\mathbf{I}_d)$. Therefore its dot product with the vector $\mathbf{w}$ results with a summation of d Gaussian vectors, each with mean 0 and variance $\sigma^2\mathbf{w}i^2\ \forall i\in[d]$. Now what remains is to prove that the value at which we want to calculate the CDF is close to 0, so that the probability of the misclassification is close to 0.5.

Recall from the analysis in Soudry et al. [2018] that asymptotically the model converges to the max-margin separator that is comprised of a weighted sum of the support vectors of the dataset (let us call this set $\mathcal{V}_B$). This means that the final weights of the model $\mathbf{w}$ is a combination of datapoints from

the first two majority distributions $\mathcal{D}_1, \mathcal{D}_2$.

$$\mathbf{w} = \sum_{i \in \mathcal{V}_B} \alpha_i \mathbf{x}_i, \tag{E.10}$$

$$\boldsymbol{\mu}_r \cdot \mathbf{w} = \sum_{i \in \mathcal{V}_B} \alpha_i (\boldsymbol{\mu}_r \cdot \mathbf{x}_i) \tag{E.11}$$

Since $\boldsymbol{\mu}_r \cdot \boldsymbol{\mu}_i = 0$, we have that $(\boldsymbol{\mu}_r \cdot \mathbf{x}_i) = \boldsymbol{\mu}_r \cdot \mathbf{z}_i + 0$, which is a mean zero random variable.

$$\boldsymbol{\mu}_r \cdot \mathbf{w} = \sum_{i \in \mathcal{V}_B} \alpha_i (\boldsymbol{\mu}_r \cdot \mathbf{z}_i), \tag{E.12}$$

$$= \sum_{i \in \mathcal{V}_B} \alpha_i \mu \sum_{2k+1 \le j \le 3k} \mathbf{z}_i j, \tag{E.13}$$

$$= \xi, \tag{E.14}$$

$$\tag{E.15}$$

where $\xi \sim \mathcal{N}(0, k\mu^2\sigma^2 \sum_{i \in \mathcal{V}_B} \alpha_i^2)$. Also, $\|\mathbf{w}\|_2^2 = \sum_{i \in \mathcal{V}_B} \alpha_i^2 \mathbf{x}_i^2$. However, $\mathbf{x}_i = \boldsymbol{\mu}_i + \mathbf{z}_i$. From Lemma 4, we know that with probability greater than $1 - \delta/6$, for every example, $\frac{d\sigma^2}{2} \le \|\mathbf{z}_i\|_2^2 \le \frac{3d\sigma^2}{2}$. By Young's inequality for products:

$$\begin{aligned}
\|\mathbf{z}_i\|_2^2 &= \left\|\mathbf{x}_i - \boldsymbol{\mu}_i\right\|_2^2, \\
&\le 2\|\mathbf{x}_i\|_2^2 + 2\left\|\boldsymbol{\mu}_2\right\|_2^2, \\
\|\mathbf{x}_i\|_2^2 &\ge \frac{1}{2}\|\mathbf{z}_i\|_2^2 - \left\|\boldsymbol{\mu}_i\right\|_2^2, \\
&\ge \frac{d\sigma^2}{4} - \left\|\boldsymbol{\mu}_i\right\|_2^2, \\
&\ge \frac{d\sigma^2}{8} \qquad \text{(since } \frac{k\mu^2}{\sigma^2} < d/nC \text{ for large C)}.
\end{aligned}$$

$$\|\mathbf{w}\|_2^2 = \sum_{i \in \mathcal{V}_B} \alpha_i^2 \mathbf{x}_i^2 \ge \frac{d\sigma^2}{8} \sum_{i \in \mathcal{V}_B} \alpha_i^2.$$

Therefore,

$$\eta = \frac{-\xi}{\sigma \|\mathbf{w}\|_2} \sim \mathcal{N}\left(0, \frac{k\mu^2}{d\sigma^2/8}\right),$$

$$\eta \sim \mathcal{N}\left(0, \frac{1}{nC}\right), \qquad \text{(since } \frac{k\mu^2}{\sigma^2} < d/nC$$

$$[(\mathbf{y}_r \mathbf{w} \cdot \mathbf{x}_r) < 0] = \Phi\left(\eta\right) \tag{E.16}$$

Using Gaussian tail bound on η, with probability at least $1 - \delta$, $\eta \le \sqrt{\frac{\log(1/\delta)}{nC}}$. Therefore, $[(\mathbf{y}_r \mathbf{w} \cdot \mathbf{x}_r) < 0] \le \Phi\left(\sqrt{\frac{\log(1/\delta)}{nC}}\right) \le \Phi\left(\frac{1}{C}\right) \approx 0.5$.

In the last step, we used the assumption (A.2) that $n \ge C \log(1/\delta)$ for some large constant C.

Remark: This analysis highlights that the meaning of being 'forgotten' for rare examples is to predict randomly. However, in case of mislabeled examples, the predicted label of the example approaches its 'true' label, irrespective of its training label.

Complex Examples The analysis for complex examples follows directly from the analysis in the case of mislabeled examples. The only difference is the magnitude of the signal to noise ratio within the group. Recall that complex examples are also sampled from majority groups. Therefore, the probability that the complex example $(\mathbf{x}, \mathbf{y})$ is misclassified at the end of training for infinite time is given by:

$$\left[\mathbf{y}\mathbf{x}^\top \mathbf{w} < 0\right] \leq \exp\left(-c\frac{||\tilde{\mu}/\lambda||^4}{d}\right) \tag{E.17}$$

This approaches 0, indicating perfect classification of complex examples from $\mathcal{D}_A$. Note that this is a complimentary case where the second split only has examples from the complex distribution.

□

Intermediate Time Analysis From the analysis in Section E, we find that all the mislabeled and rare examples are forgotten by the time we train for $t \to \infty$ iterations. Since examples from complex subgroups are not forgotten even at infinite time training, we skip analysis for those examples in this subsection. Our goal is to show that there exists a time T' such that with high probability, the model forgets all the mislabeled examples, but still correctly classifies all the rare examples. To show this, we will track the accuracy of inputs in the first data split, as we train on the examples in the second split.

The model output for any sample $(\mathbf{x}_i, \mathbf{y}_i) \in \mathcal{D}_A$ is given by $\mathbf{w}(t)^\top \mathbf{x}_i$, and the prediction is considered to be correct if $\text{sign}(\mathbf{w}(t)^\top \mathbf{x}_i) = \mathbf{y}_i$, or if $\mathbf{w}(t)^\top \mathbf{x}_i \mathbf{y}_i > 0$. From hereon, we will use the notation $\gamma_i^t = \mathbf{w}(t)^\top \mathbf{x}_i \mathbf{y}_i$.

Recall that we considered that there is one example from both the mislabeled and rare example category in $\mathcal{D}_A$. We will denote these data points by $(\mathbf{x}_m, \mathbf{y}_m) \in \mathcal{M}$, and $(\mathbf{x}_r, \mathbf{y}_r) \in \mathcal{A}$ respectively. Without loss of generality, we will assume that $(\mathbf{x}_m, \neg\mathbf{y}_m)$ was sampled from $\mathcal{D}_1$ in the mixture of distributions $\mathcal{D}$. All the examples in the second split $\mathcal{D}_B$ sampled from the same distribution are given by $\mathcal{D}_{B,1} \subset \mathcal{D}_B$. The remaining examples are in the set denoted by $\mathcal{D}_{B,\neg 1} \subset \mathcal{D}_B$.

From the learning time training dynamics, we know that the initialization of the weights for the second round of training is given by:

$$\mathbf{w}_B(0) = \hat{\mathbf{w}}_A \log T + \rho_A(T). \tag{E.18}$$

Now, from the representer theorem, we can decompose the model weights at any iteration of second-split training as follows,

$$\mathbf{w}_B(t) = \hat{\mathbf{w}}_A \log T + \rho_A(T) + \sum_{j \in \mathcal{D}_B} \beta_j \mathbf{y}_j \mathbf{x}_j. \tag{E.19}$$

Note that we introduce an additional term $\mathbf{y}_j$ in the decomposition using representer theorem, but this can be done without loss of generality since $\mathbf{y}_j \in \{-1, +1\}$. This helps us in ensuring that each β_j is non-negative as shown in Lemma 6.

From Lemma 8, we know that there exists a bounded time T' when the mislabeled example prediction flips. We will denote μ as the coordinate-wise mean for the k-signal dimensions in the vector $\boldsymbol{\mu}_g$ in the discussion that follows. Let us define $\Delta_t = \frac{\sum_{j \in \mathcal{D}_{B,1}} \beta_j(t)}{\sum_{j \in \mathcal{D}_B} \beta_j(t)}$, and $\Delta = \max_t \Delta_t$. For a symmetric distribution with two majority subgroups with the opposite label, we would expect this value to be close to 0.5. Now, we present formal version of Theorem 4 from the main paper.

Theorem 6 (Intermediate-Time Forgetting)**.**

Under the distribution outlined in Appendix E with assumptions A.1, A.2, A.3, whenever $\mathcal{D}_A$ is separable, when fine-tuning on the second dataset split $\mathcal{D}_B$ with sufficiently small learning rate, there exists some bounded time T' when

1. $_{(\mathbf{x}_m,\mathbf{y}_m)\in\mathcal{D}_A}[\mathbf{y}_m \neq \mathbf{w}(T')\cdot\mathbf{x}_m] \geq 1-c_0\exp\left(\frac{-k^2\mu^2\Delta^2}{c\sigma^2 d}\right)-c_1\exp(-cd)$
2. $_{(\mathbf{x}_r,\mathbf{y}_r)\in\mathcal{D}_A}[\mathbf{y}_r = \mathbf{w}(T')\cdot\mathbf{x}_r] \geq 1-c_0\exp\left(\frac{-k^2\mu^2\Delta^2}{c\sigma^2 d}\right)-c_1\exp(-cd)$

for absolute constants c_0, c_1, c and $(\mathbf{x}_m,\mathbf{y}_m)\in\mathcal{M}, (\mathbf{x}_r,\mathbf{y}_r)\in\mathcal{A}$ respectively.

If the fraction of dimensions that contain the signal, k/d, is considered fixed then both the first and second term above exponentially decay with a factor of d. This suggests that increasing overparametrization leads to a higher likelihood of the phenomenon of intermediate time forgetting—there exists an epoch when the prediction of mislabeled example is flipped but the rare examples are still correctly predicted with high probability.

In what follows, the key idea is to show that at a time when the prediction of the mislabeled example is incorrect with high probability, the predictions of the rare examples is correct with high probability.

Proof. From Lemma 7, we know that $(\mathbf{x}_m,\mathbf{y}_m)$ and $(\mathbf{x}_r,\mathbf{y}_r)$ are support vectors for $\mathcal{D}_A$. Hence, $\mathbf{y}_m\mathbf{x}_m^\top\hat{\mathbf{w}}_A = \mathbf{y}_r\mathbf{x}_r^\top\hat{\mathbf{w}}_A = 1$. We will use this fact to find the distribution for $_m^t$, $_r^t$.

Let $\mathbf{x}_j i$ denote the i^{th} dimension of the j^{th} datapoint in the second split. Recall that the second split comprises of two majority subgroups. k dimensions of the input vector contain the true signal for class prediction. The dimensions $\{1\dots k\}, \{k+1\dots 2k\}, \{2k+1\dots 3k\}$ are the predictive dimensions for the majority group 1, 2 and that for the rare example from dataset $\mathcal{D}_A$. Therefore, $\mathbf{x}_j i = \mu + \mathbf{z}_j i$, if i is in the predictive dimensions, otherwise $\mathbf{x}_j i = \mathbf{z}_j i$ where $\mathbf{z}_j i \sim \mathcal{N}(0,\sigma^2)$. To make notation simple, we will refer to $\beta_j(t)$ by using β_j.

Remark: We do not perform input transformation to prove the following results.

Mislabeled Example The prediction on the mislabeled point times the given label can be written as:

$$_m^t = \mathbf{y}_m\mathbf{x}_m^\top\mathbf{w}_B(t) = \mathbf{y}_m\mathbf{x}_m^\top\left(\hat{\mathbf{w}}_A\log T + \rho_A(T) + \sum_{j\in\mathcal{D}_B}\beta_j\mathbf{y}_j\mathbf{x}_j\right),$$
$$= \log T + c_m + \sum_{j\in\mathcal{D}_B}\beta_j(\mathbf{y}_m\mathbf{x}_m^\top\mathbf{y}_j\mathbf{x}_j) \qquad \text{(usi}$$
(E.20)

where $c_m = \mathbf{y}_m\mathbf{x}_m^T\rho_A(T)$ is a residual term that does not continue grow with T (Theorem 9 [Soudry et al., 2018]).

Without loss of generality, assume that the mislabeled example $\mathbf{x}_m$ belongs to majority group 1 (true label = 1), and was originally labeled such that $\mathbf{y}_m$ = -1 in the first-split dataset.

$$\sum_{j\in\mathcal{D}_B}\beta_j(\mathbf{y}_m\mathbf{x}_m^\top\mathbf{y}_j\mathbf{x}_j) = \sum_{j\in\mathcal{D}_{B,1}}\beta_j(\mathbf{y}_m\mathbf{x}_m^\top\mathbf{y}_j\mathbf{x}_j) + \sum_{j\in\mathcal{D}_{B,\neg 1}}\beta_j(\mathbf{y}_m\mathbf{x}_m^\top\mathbf{y}_j\mathbf{x}_j)$$
$$= -\sum_{j\in\mathcal{D}_{B,1}}\beta_j(\mathbf{x}_m^\top\mathbf{x}_j) + \sum_{j\in\mathcal{D}_{B,\neg 1}}\beta_j(\mathbf{x}_m^\top\mathbf{x}_j). \tag{E.21}$$

Now we can use the distribution properties of the dataset to further simplify per dimension and aggregate the sum across all examples.

$$\sum_{j\in\mathcal{D}_{B,1}}\beta_j(\mathbf{x}_m^\top\mathbf{x}_j) = \mathbf{x}_m^\top\sum_{j\in\mathcal{D}_{B,1}}\beta_j\mathbf{x}_j = \mathbf{x}_m^\top\mathbf{x}_{\mathcal{D}_{B,1}}, \tag{E.22}$$

where $\mathbf{x}_{\mathcal{D}_{B,1}}i = \mu\,\mathbb{1}(i\in\{1\dots k\})\sum_{\mathcal{D}_{B,1}}\beta_j + \mathbf{z}_{\mathcal{D}_{B,1}}i$, where $\mathbf{z}_{\mathcal{D}_{B,1}}i \sim \mathcal{N}(0,\sigma^2\sum_{j\in\mathcal{D}_{B,1}}\beta_j^2)$. Now, we will add up the dot product dimension wise. Let us call $B_1 = \sum_{\mathcal{D}_{B,1}}\beta_j$ and $B_1^v = \sum_{j\in\mathcal{D}_{B,1}}\beta_j^2$. Then $\mathbf{x}_{\mathcal{D}_{B,1}}i \sim \mathcal{N}(\mu\,\mathbb{1}(i\le k)\cdot B_1, \sigma^2 B_1^v)$.

$$\mathbf{x}_m^\top\mathbf{x}_{\mathcal{D}_{B,1}} = \mu\sum_k \mathbf{z}_{\mathcal{D}_{B,1}}i + \mu B_1\sum_k \mathbf{z}_m i + k\cdot\mu^2\cdot B_1 + \sum_d \mathbf{z}_{\mathcal{D}_{B,1}}i\cdot\mathbf{z}_m i$$
$$= \mu\cdot\alpha_1 + \sum_d \mathbf{z}_{\mathcal{D}_{B,1}}i\cdot\mathbf{z}_m i$$
$$\mathbf{x}_m^\top\mathbf{x}_{\mathcal{D}_{B,\neg 1}} = \mu\cdot\alpha_2 + \sum_d \mathbf{z}_{\mathcal{D}_{B,\neg 1}}i\cdot\mathbf{z}_m i \tag{E.23}$$

where $\alpha_1 \sim \mathcal{N}(k\cdot\mu\cdot B_1, k\cdot\sigma^2(B_1^v + B_1^2))$ and $\alpha_2 \sim \mathcal{N}(0, k\cdot\sigma^2(B_2^v + B_2^2))$. Notice that in the first step, the first three terms are independent of each other and can be added directly to obtain a new random variable using the independence condition, however, the last term is dependent on the first two. In the final step, we also add the terms for the dot product corresponding to the opposite group.

This gives us the overall expression as follows:

$$\sum_{j\in\mathcal{D}_B}\beta_j(\mathbf{y}_m\mathbf{x}_m^\top\mathbf{y}_j\mathbf{x}_j) = \mu\cdot(\alpha_2-\alpha_1) + \sum_d(\mathbf{z}_{\mathcal{D}_{B,\neg 1}}i - \mathbf{z}_{\mathcal{D}_{B,1}}i)\cdot\mathbf{z}_m$$
$$= \mu\cdot\alpha_m + \sum_d \mathbf{z}_{\mathcal{D}_B}i\cdot\mathbf{z}_m i - k\cdot\mu^2\cdot B_1, \tag{E.24}$$

where $\alpha_m \sim \mathcal{N}(0, k\cdot\sigma^2(B^v + B_1^2 + B_2^2))$ and $\mathbf{z}_{\mathcal{D}_B}i \sim \mathcal{N}(0,\sigma^2 B^v)$, since $(B_1^v + B_2^v = \sum_{j\in\mathcal{D}_B}\beta_j^2 = B^v)$.

$$^t_m = \log T + c_m + \xi_m - k\cdot\mu^2\cdot B_1;\ \text{s.t. } \xi_m = \mu\cdot\alpha_m + \sum_d \mathbf{z}_{\mathcal{D}_B} \tag{E.25}$$

Rare Example Following the analysis of the mislabeled example, we can similarly find an expression for ^t_r on the rare example as follows:

$$^t_r = \mathbf{y}_r\mathbf{x}_r^\top\mathbf{w}_B(t) = \mathbf{y}_r\mathbf{x}_r^\top(\hat{\mathbf{w}}_A\log T + \rho_A(T) + \sum_{j\in\mathcal{D}_B}\beta_j\mathbf{y}_j\mathbf{x}_j),$$
$$= \log T + c_r + \sum_{j\in\mathcal{D}_B}\beta_j(\mathbf{y}_r\mathbf{x}_r^\top\mathbf{y}_j\mathbf{x}_j) \qquad \text{(using l} \tag{E.26}$$

Following the same procedure as for mislabeled example, we get the overall expression as follows:

$$\sum_{j\in\mathcal{D}_B}\beta_j(\mathbf{y}_r\mathbf{x}_r^\top\mathbf{y}_j\mathbf{x}_j) = \mu\cdot\alpha_r + \sum_d \mathbf{z}_{\mathcal{D}_B}i\cdot\mathbf{z}_r i, \tag{E.27}$$

where $\alpha_r \sim \mathcal{N}(0, k \cdot \sigma^2(B^v + B_1^2 + B_2^2))$ and $\mathbf{z}_{\mathcal{D}_B} i \sim \mathcal{N}(0, \sigma^2 B^v)$.

$$\overset{t}{r} = \log T + c_r + \xi_r, \text{ s.t. } \xi_r = \mu \cdot \alpha_r + \sum_d \mathbf{z}_{\mathcal{D}_B} i \cdot \mathbf{z}_r i. \tag{E.28}$$

Combining both cases Recall that $\beta_j > 0$ for all j. Therefore, $B^2 = (\sum_{j \in \mathcal{D}_B} \beta_j)^2 > B^v = \sum_{j \in \mathcal{D}_B} \beta_j^2$. Based on the analysis of Soudry et al. [2018], we know that $\mathbf{w}_B(t)$ grows as fast as $O(\log t)$. Therefore, both $B_1 = O(\log t)$, and $B = O(\log t)$ (must grow at most as fast as that). From the problem definition, $B_1 \leq \Delta B$, where $\Delta \in [0, 1]$.

Our goal now is to find an epoch t during the second-split training when the rare example is correctly classified with high probability, and the mislabeled example is incorrectly classified with high probability. The maxima is achieved when $\overset{t}{r} \approx -\overset{t}{m}$. We assume that the step sizes are sufficiently small such that we can achieve this condition. Moreover, $c_r, c_m L^2(\Omega) \log T$. Hence, the condition is to find t such that $\xi_r + \log T > 0$ and $\xi_m + \log T < k \cdot \mu^2 \cdot B_1$ with high probability. By symmetry, we must find $\left(|\xi_m| < k \cdot \mu^2 \cdot B_1/2\right)$.

$$\left(|\xi_m| > k \cdot \mu^2 \cdot \frac{B_1}{2}\right) \leq \left(|\xi_1| > k \cdot \mu^2 \cdot \frac{B_1}{4}\right) + \left(|\xi_2| > k \cdot \mu^2 \cdot \frac{B_1}{4}\right), \tag{E.29}$$

where $\xi_1 = \mu \cdot \alpha_m \sim \mathcal{N}(0, \mu^2 \sigma^2 k(B^v + B_1^2 + B_2^2)$ and $\xi_2 = \sum_d \mathbf{z}_{\mathcal{D}_B} i \cdot \mathbf{z}_r i$. For the first term, we can directly use the Gaussian tail bound using Chernoff method.

$$\begin{aligned}
\left(|\xi_1| > k \cdot \mu^2 \cdot \frac{B_1}{4}\right) &\leq 2 \exp \frac{-k^2 \mu^4 B_1^2}{32 \mu^2 \sigma^2 d(B^v + B_1^2 + B_2^2)}, \\
&\leq 2 \exp \frac{-k^2 \mu^4 B_1^2}{32 \mu^2 \sigma^2 d(2B^2)}, \\
&\leq 2 \exp \frac{-k \mu^2 \Delta^2}{c_1 \sigma^2}.
\end{aligned} \tag{E.30}$$

Now, for the second term, using Lemma 3 we have that.

$$\begin{aligned}
\left(|\xi_2| > k \cdot \mu^2 \cdot \frac{B_1}{4}\right) &\leq 2 \exp \frac{-k^2 \mu^4 B_1^2}{c_2 \sigma^4 d B^v} + c_1 \exp(-cd), \\
&\leq 2 \exp \frac{-k^2 \mu^2 \Delta}{c_2 \sigma^2 d} + c_1 \exp(-cd),
\end{aligned} \tag{E.31}$$

since $\frac{\mu}{\sigma} > 1$. Finally, combining both the cases we have that

$$\left(|\xi_m| > k \cdot \mu^2 \cdot \frac{B_1}{2}\right) \leq c_0 \exp \frac{-k^2 \mu^2 \Delta^2}{c \sigma^2 d} + c_1 \exp(-cd), \tag{E.32}$$

This shows that as the dimensionality of the dataset increases, the likelihood of prediction on mislabeled examples being flipped while the rare examples retain their prediction increases exponentially. This concludes the proof of Theorem 6. □

Concentration Inequalities and Additional Lemmas To make our work self-contained, we supplement the reader with additional Lemmas and Theorems that are helpful tools

for proving the theorems in this work. We restate versions of the Hoeffding and Bernstein inequalities as in Chatterji and Long [2021].

Lemma 2 (Soudry et al. [2018], Theorem 3)**.** *For any linearly separable dataset $\mathcal{D}_A$ and for all small enough step-sizes α, we have*

$$\frac{\mathbf{w}_A}{\|\mathbf{w}_A\|} = \lim_{t\to\infty} \frac{\mathbf{w}^{(t)}}{\|\mathbf{w}^{(t)}\|}.$$

Theorem 7 (General Hoeffding's Inequality)**.** *Let $\theta_1, \ldots, \theta_m$ be independent mean-zero sub-Gaussian random variables and $a = (a_1, \ldots, a_m) \in \mathbb{R}^m$. Then, for every $t > 0$, we have*

$$\left[\left|\sum_{i=1}^{m} a_i\theta_i\right| \geq t\right] \leq 2\exp\left(\frac{-c_0\, t^2}{K^2\,\|a\|_2^2}\right),$$

where $K = \max_i \|\theta_i\|_{\psi_2}$ and c is an absolute constant.

In our case, since we deal with Gaussian random variables, $\|\theta\|_{\psi_2}$ (sub-Gaussian norm) is same as the variance of the random variable. That is, $\theta \sim \mathcal{N}(0,\sigma^2) \implies \|\theta\|_{\psi_2} = K = \sigma$.

Theorem 8 (Bernstein Inequality)**.** *For independent mean-zero sub-exponential random variables $\theta_1, \ldots, \theta_m$, for every $t > 0$, we have*

$$\left[\left|\sum_{i=1}^{m} \theta_i\right| \geq t\right] \leq 2\exp\left(-c_1 \min\left\{\frac{t^2}{\sum_{i=1}^m \|\theta_i\|_{\psi_1}^2}, \frac{t}{\max_i \|\theta_i\|_{\psi_1}}\right\}\right)$$

where c_1 is an absolute constant.

In our case, let $\mathbf{x}_i \sim \mathcal{N}(0,\sigma^2)$ be independent Gaussian random variables, then for $Z = \sum_i^d X_i^2$,

$$\left(\left(|Z - d\sigma^2| \geq t\right) \leq 2\exp\left(-\frac{c_1}{\sigma^2}\min\left\{\frac{t^2}{d\sigma^2}, t\right\}\right)\right.$$

Lemma 3 (Gaussian Product)**.** *Let $\mathbf{z}_1 \sim \mathcal{N}(0,\sigma_1^2\mathbf{I}_d)$, $\mathbf{z}_2 \sim \mathcal{N}(0,\sigma_2^2\mathbf{I}_d)$ be independent multivariate Gaussian random variables. Then,*

$$[|\mathbf{z}_1 \cdot \mathbf{z}_2| > t] \leq 2\exp\left(\frac{-c\, t^2}{d\sigma_1^2\sigma_2^2}\right) + 2\exp\left(-c_0 d\right)$$

Proof. The proof of this lemma is a simplified version of the proof for Lemma 20 in Chatterji and Long [2021]. First, consider $\mathbf{z}_2$ to be fixed, and only $\mathbf{z}_1$ to be random. Then from Theorem 7 we have

$$\left[\left|\sum_{i=1}^{d} \mathbf{z}_2 i \cdot \mathbf{z}_1 i\right| \geq t\right] \leq 2\exp\left(\frac{-c\, t^2}{\sigma_1^2\,\|\mathbf{z}_2\|_2^2}\right).$$

Also, adapting Theorem 8 such that $Z = \sum_i^d X_i^2 = \|\mathbf{z}_2\|_2^2$, and setting $t = d\sigma^2$

$$\left(\left(\left|\|\mathbf{z}_2\|_2^2 - d\sigma_2^2\right| \geq t\right) \leq 2\exp\left(-\frac{c_0}{\sigma^2}\min\left\{\frac{t^2}{d\sigma_2^2}, t\right\}\right),\right.$$

$$\left(\left(\|\mathbf{z}_2\|_2^2 \geq 2d\sigma_2^2\right) \leq 2\exp\left(-c_0 d\right).\right.$$

Coming back to the initial problem and again considering both $\mathbf{z}_1, \mathbf{z}_2$ to be random variables, we have

$$[|\mathbf{z}_1 \cdot \mathbf{z}_2| \geq t] \leq \left[|\mathbf{z}_1 \cdot \mathbf{z}_2| \geq t \;\middle|\; \|\mathbf{z}_2\|_2^2 \leq 2d\sigma_2^2\right] + \left[\|\mathbf{z}_2\|_2^2 > 2d\sigma_2^2\right] \tag{E.33}$$

$$\leq 2\exp\left(\frac{-c\, t^2}{d\sigma_1^2\sigma_2^2}\right) + 2\exp\left(-c_0 d\right). \tag{E.34}$$

□ □

Corollary 1 (Chatterji and Long [2021], Lemma 20)**.** *There is a $c \geq 1$ such that, for all large enough C, with probability at least $1 - \delta/6$, for all $i \neq j \in [n]$,*

$$|\mathbf{z}_i \cdot \mathbf{z}_j| < c\left(\sigma^2\sqrt{d\log(n/\delta)}\right).$$

Lemma 4 (Gaussian Square)**.** *There is a $c \geq 1$ such that, for all large enough C, with probability at least $1 - \delta/6$, for all $k \in [n]$,*

$$\frac{d\sigma^2}{2} \leq \|\mathbf{z}_k\|_2^2 \leq \frac{3d\sigma^2}{2}.$$

Proof. Recall that $\mathbf{x}_k = \boldsymbol{\mu}_k + \mathbf{z}_k$, where $\mathbf{z}_k \sim \mathcal{N}(0, \sigma^2\mathbf{I}_d)$. Then,

$$\|\mathbf{z}_k\|_2^2 = \sum_i^d \mathbf{z}_k i^2 = Z$$

Adapting Theorem 8, and setting $t = \lambda d\sigma^2$ with $0 < \lambda < 1$

$$\left(|\|\mathbf{z}_2\|_2^2 - d\sigma^2| \geq t\right) \leq 2\exp\left(-\frac{c_1}{\sigma^2}\min\left\{\frac{t^2}{d\sigma^2}, t\right\}\right),$$

$$\left(|\|\mathbf{z}_2\|_2^2 - d\sigma^2| \geq \lambda d\sigma^2\right) = 2\exp\left(-\frac{c_1}{\sigma^2}\min\left\{\lambda^2 d\sigma^2, \lambda d\sigma^2\right\}\right),$$

$$= 2\exp\left(-c_1\lambda^2 d\right). \quad \text{(since } 0 < \lambda < 1\text{)}$$

Recall that $d \geq C\log(n/\delta)$. We can set $\lambda = 1/2$ so that, $\|\mathbf{z}_2\|_2^2 > d\sigma^2/2$ with probability at least $1 - \delta/6n$ (for large enough C). Taking a union bound over all examples gives us the desired result.

Note that we can get closer to $d\sigma^2$ by chosing an appropriately higher value of C.

Lemma 5 (Dataset Separability)**.** *With probability at least $1-\delta$ over random samples of dataset $\mathcal{D}_A$, samples $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ are linearly separable.*

Proof. We will show that there exists a set of weights that with high probability correctly classify each example in the dataset. Let $\mathbf{x}_i = \boldsymbol{\mu}_i + \mathbf{z}_i$ for every data point in $\mathcal{D}$. From assumptions, our dataset ($\mathcal{D}_A$) contains $\mathbf{z}_r, \mathbf{z}_m$ that belong to rare and mislabeled groups. Rest all points are denoted by $\mathbf{z}_k$. Consider the classifier $\mathbf{w} = \sum_i^n \mathbf{y}_i\mathbf{x}_i$. Then,

Case 1: Rare Example $(\mathbf{x}_r, \mathbf{y}_r)$

$$\begin{aligned}
\mathbf{y}_r\mathbf{w}\cdot\mathbf{x}_r &= \sum_j \mathbf{y}_j\mathbf{y}_r\mathbf{x}_j\cdot\mathbf{x}_r, \\
&= \mathbf{x}_r\cdot\mathbf{x}_r + \sum_{i\neq j}\mathbf{y}_r\mathbf{y}_j\mathbf{x}_r\cdot\mathbf{x}_j, \\
&= \boldsymbol{\mu}_r\cdot\boldsymbol{\mu}_r + \mathbf{z}_i\cdot\mathbf{z}_i + \sum_{r\neq j}\mathbf{y}_r\mathbf{y}_j\mathbf{z}_r\cdot\mathbf{z}_j, && \text{(since } \boldsymbol{\mu}_r\cdot\boldsymbol{\mu} \\
&= k\mu^2 + \mathbf{z}_i\cdot\mathbf{z}_i + \sum_{r\neq j}\mathbf{y}_r\mathbf{y}_j\mathbf{z}_r\cdot\mathbf{z}_j, \\
&\geq 0 + d\sigma^2/2 - c_1 n\sqrt{d\sigma^2\log(n/\delta)} && \text{(using Lemm} \\
&> 0, \\
&&& \text{for } d \geq Cn^2\log(n/\delta).
\end{aligned}$$

Case 2: Mislabeled Example $(\mathbf{x}_m, \mathbf{y}_m)$ Without loss of generality, assume that the mislabeled example is sampled from the distribution $\mathcal{D}_1$, and the set of all correctly labeled examples in the dataset $\mathcal{D}_A$ from this distri-

bution be $\mathcal{D}_{A,1}$.

$$\begin{aligned}
\mathbf{y}_m \mathbf{w} \cdot \mathbf{x}_m &= \sum_j \mathbf{y}_j \mathbf{y}_m \mathbf{x}_j \cdot \mathbf{x}_m, \\
&= \mathbf{x}_m \cdot \mathbf{x}_m + \sum_{i \neq j} \mathbf{y}_m \mathbf{y}_j \mathbf{x}_m \cdot \mathbf{x}_j, \\
&= \boldsymbol{\mu}_m \cdot \boldsymbol{\mu}_m + \mathbf{z}_i \cdot \mathbf{z}_i - \sum_{j \in \mathcal{D}_{A,1}} \boldsymbol{\mu}_m \cdot \boldsymbol{\mu}_m + \sum_{m \neq j} \mathbf{y}_m \mathbf{y}_j \mathbf{z}_m \cdot \mathbf{z}_j, \\
&\geq 0 + d\sigma^2/2 - n_1 k\mu^2 - c_1 n\sqrt{d \log(n/\delta)}, \\
&\geq d\sigma^2/2 - nk\mu^2 - c_1 n\sigma^2\sqrt{d \log(n/\delta)}, \\
&> 0
\end{aligned}$$

for $d \geq C \max\{n^2 \log(n/\delta), nk\mu^2/\sigma^2\}$.

Case 3: Majority Example $(\mathbf{x}_i, \mathbf{y}_i)$ The case of majority examples directly follows from the case of mislabeled examples. Rather than having a negative summation over the mean vector for n examples (in line 3), we will have a positive summation because the true label will match the label of the rest of the examples in the subset $\mathcal{D}_{A,1}$. This will make the expected value of the dot product even larger.

Remark: Since the first split dataset $\mathcal{D}_A$ is separable, it directly follows that the second split dataset $\mathcal{D}_B$ is also separable since it does not have any mislabeled and rare examples. □

Lemmas for Theorem 6

Lemma 6 (Sign of β)**.** *$\beta_j \geq 0$ for all j in Equation E.19.*

Proof. Analyzing the steps of gradient descent, we have:

$$\begin{aligned}
\dot{\mathbf{w}}(t) &= -\nabla \mathcal{L}(\mathbf{w}(t)) \\
&= \sum_{j \in \mathcal{D}_B} \exp\left(-\mathbf{y}_j \mathbf{x}_j^\top \mathbf{w}(t)\right)(\mathbf{y}_j \mathbf{x}_j^\top) \\
\mathbf{w}(t) - \mathbf{w}(0) &= \sum_{j \in \mathcal{D}_B} \mathbf{y}_j \mathbf{x}_j^\top \underbrace{\int_0^t \exp\left(-\mathbf{y}_j \mathbf{x}_j^\top \mathbf{w}(t)\right) dt}_{\beta_j(t)}, \\
&\text{(since } \mathbf{w}(0) = 1\text{, using Lemma 4 and Corollary 1)}
\end{aligned} \tag{E.35}$$

Hence, $\beta_j \geq 0 \; \forall \; j$. □

Lemma 7 (Support Vectors)**.** *If dataset $\mathcal{D}_A$ is separable, then $(\mathbf{x}_m, \mathbf{y}_m)$ and $(\mathbf{x}_r, \mathbf{y}_r)$ are support vectors for $\mathcal{D}_A$.*

Proof. We will prove by contradiction. From our assumption, $(\mathbf{x}_m, \mathbf{y}_m)$ and $(\mathbf{x}_r, \mathbf{y}_r)$ are the only mislabeled and rare examples in the first-split $\mathcal{D}_A$ from their respective subgroups. Let us assume that they are not support vectors. Then, we can directly follow from the Asymptotic Analysis in Subsection E that the probability of correct classification of the rare example is 0.5 and for the mislabeled example approaches 0 as the model is trained for infinite time. But we know that the model achieves 100% accuracy on the training set $\mathcal{D}_A$ with weights $\mathbf{w}_A(T)$. Hence, this is a contradiction, and $(\mathbf{x}_m, \mathbf{y}_m)$, $(\mathbf{x}_r, \mathbf{y}_r)$ must be support vectors for the original classification problem. □

Lemma 8 (Bounded Time Prediction)**.** *With probability at least $1 - \delta$, there exists a bounded*

time $T' = O(T)$ when the mislabeled examples are incorrectly classified with high probability.

Proof. In Subsection E we have shown that at infinite time training, the model misclassifies mislabeled examples with high probability. Then, the existence of bounded time weights for which the prediction of mislabeled examples is flipped directly follows from the proof of Proposition 1 applied to the result of Theorem 5. □

Experimental Results

Experimental Setup

Architectures We perform experiments using four different model architectures—LeNet, ResNet-9 [Baek, 2019], ResNet-50, and Bert-base-cased [Devlin et al., 2018]. Comparisons with model architectures are used in analysis of stability of the SSFT metric. For other numbers reported in tables and plots, we use the ResNet-9 model, unless otherwise stated.

Optimizer We experiment with three different learning rate scheduling strategies—cyclic learning rate schedule, cosine learning rate, and step decay learning rate. We test for two values of peak learning rate—0.1 and 0.01. All the model are trained using the SGD optimizer with weight decay 5e-4 and momentum 0.9, apart from the comparison with optimizers in Appendix E where we also experiment with the Adam optimizer.

Training Procedure We train for a maximum of 100 epochs or until we have 5 epochs of 100% training accuracy. We first train on $\mathcal{D}_A$, and then using the pre-initialized weights from stage 1, train on $\mathcal{D}_B$ with the same learning parameters. All experiments can be performed on a single RTX2080 Ti.

Image Datasets In the main paper we present visualizations of training examples from the MNIST dataset based on which quadrant they lie on in the learning-forgetting graph. Here, we complement our findings by showing visualizations for the CIFAR-10 dataset. We note that CIFAR-10 dataset provide many different types of visibility patterns within the same class. Hence, examples may be learnt late due to belonging to a rare visibility pattern. In Figure E.1, we see that the examples that were learnt earliest and never forgotten have similar visibility patterns—for instance all the trucks have a similar perspective. On the contrary, as we move to the first quadrant with examples that were learnt late but never forgotten, we see that all the examples are true to their semantic class, but these visibility patterns occur rarely. Finally, we also analyze the visualizations based on examples that were forgotten during the course of second-split training. While in the case of MNIST dataset, SSFT was able to remove the mislabeled examples well, we see that CIFAR-10 offers more challenges because examples may be ambiguous because of other reasons and may be forgotten owing to the model using spurious features.

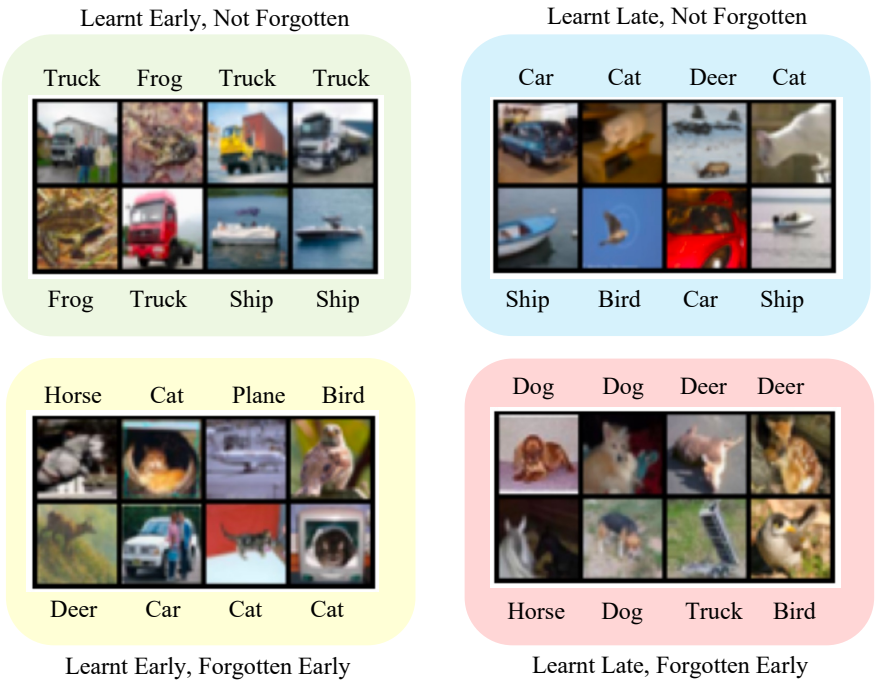


Figure E.1: Examples from the CIFAR-10 dataset grouped based on their learning and forgetting time.

Ablation Studies

We detail the experimental setup used to conduct our ablation studies directed towards understanding the learning and forgetting dynamics of rare and complex examples respectively.

Rare Examples The experiments to show the rate of learning for rare examples are inspired by the singleton hypothesis as proposed by Feldman [2020a]. The hypothesis was inspired by a long-tailed distribution of visibility patterns in the person and bus category of the PASCAL dataset. For example, the dataset contains many buses with the front visible, but very few buses that were captured from the rear or the side, and even fewer buses whose view is occluded by the presence of other objects infront of them. (Refer to Figure 1 in their work for more details.) In our work, we first attempted to leverage the same training set-up with the provided visibility patters. However, we noted that there wasn't a strong correlation between the frequency of an example's visibility pattern, and the rate at which it was learnt. We hypothesize that this is because there are other factors of example hardness that may make an example be learnt slowly or fast (such as complexity, as detailed in the next paragraph). This can lead to an example being learnt fast if it has a simple pattern yet occurs rarely. Especially when there are only $O(1)$ samples from a given sub-group (based on the visibility pattern), we can not make any claims based on singleton correlation alone.

Hence, in order to distill the frequency of occurrence of an example with other con-

founders that may influence its training-time, we created a long-tailed dataset from the CIFAR-100 dataset. CIFAR-100 is a dataset of 100 object classes, which can be further grouped into 20 super-classes. For instance, examples from categories *maple, oak, palm, pine, willow* all belong to the 'superclass' of *trees*. Similar division of 5 sub-classes is provided in the datasets for each of the super-classes. Each class contains 500 training examples and the overall dataset has 50,000 training examples.

As a first step towards creating a long-tailed dataset, we assign a fixed frequency ordering within the subgroups of a superclass. The most frequent subgroup has 500 examples in the training set, for the next most frequent subgroup, we randomly select 250 examples from the training set, and so on until the last sub group with 31 examples in the training set. This means that there are exactly 20 sub-groups in the final dataset with {500,250,125,62,31} examples respectively. Irrespective of the class number, the task is to predict the corresponding superclass, that is, we reduce the problem to a 20-class classification problem. However, we track the learning and forgetting dynamics of examples from each of the 100 sub-groups separately, based on their group frequency. To remove any other confounders of example hardness, we (i) randomize the group frequency ordering of the sub-groups within a superclass (in case some classes are harder to learn than the others); and (ii) randomize the examples that were selected based on the group size (in case some examples were ambiguous or hard). We further split the dataset into two IID partitions to analyze the learning time and SSFT, and average the results over 20 random runs of the experiment. Experimental results are detailed in the main paper.

Complex Examples Prior works advocating for, and understanding the simplicity bias [Shah et al., 2020] have operationalized the notion of simplicity via the complexity of hypothesis class required to learn the distribution that a complex example may be sampled from. In particular, Shah et al. [2020] construct a synthetic dataset with MNIST and CIFAR-10 images vertically stacked on top of each other—with the part with MNIST images corresponding to the part of the combined image with *simpler* features, and the part with CIFAR-10 images corresponding to the part with *complex* features. They show that the model almost completely relies on the part of the image containing the MNIST digit even when it is less predictive of the true label. Inspired by this argument about the simplicity of features, we create a dataset that has the the union of images from the MNIST and the CIFAR-10 dataset. More specifically, we select classes from the MNIST dataset corresponding to digits $\{0, 1, 2, 3\}$, and classes from the CIFAR-10 dataset corresponding to {horses, airplanes, dog, frog} and label them from $\{0, 1, 2, 3\}$. This means that the model associates the label 0 to both the digit 0 and airplane class. The attempt of this experiment is to draw the link between the simplic-

ity bias and the rate of learning. Experimental results are provided in the main paper.

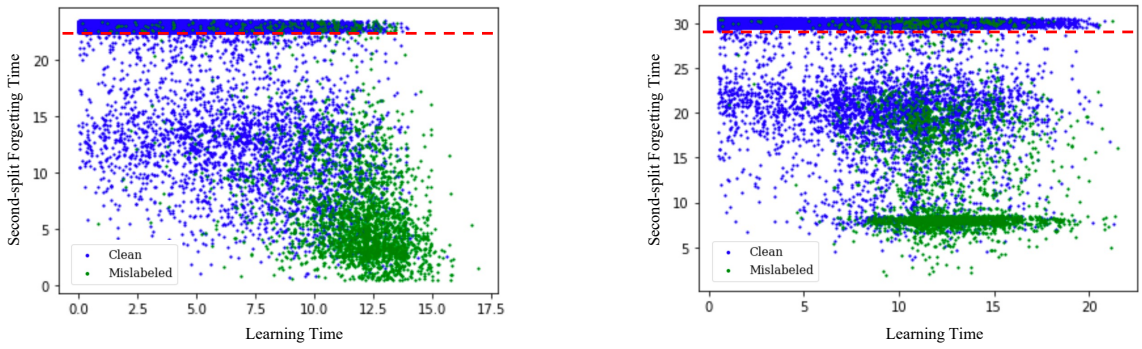
Stability of our metric

Stability across architectures The forgetting of examples is a property of both the dataset and the model architecture. As a result, we find that just like the learning time, the forgetting time has a lower correlation between architectures. The average pearson correlation between the ResNet-9 and ResNet-50 models is 0.62 in case of the CIFAR10 dataset. However, we note that the most forgotten examples generalize across datasets. That is, the average pearson correlation between the bottom 10% examples of the dataset is 0.87. This highlights how the forgetting metric is good for finding misaligned examples in the dataset, since they are not a property of the model architecture. We suspect that among the examples that are infrequently forgotten, the model campacity and other inductive biases of the model architecture may have a role in driving the average pearson correlation low.

Stability across optimizers Jiang et al. [2020] showed that changing the learning optimizer from SGD to Adam can lead to a significant change in the learning rate of examples from different levels of hardness (based on their regularity metric). More specifically, they find that examples with a low consistency score (closely correlated with learning speed) also get learnt fast when using the Adam optimizer. This suggests that using an optimizer like Adam at training time may have an impact on the ability of learning time based metrics to separate examples. In Figure E.2, we contrast the ability of forgetting and learning time based metrics for identifying label noise when using the SGD and Adam optimizers. When using an optimizer such as SGD, the mislabeled samples are learnt slower than a large fraction of the training examples, and the learning time metric offers some degree of separation between the clean and mislabeled examples. However, when we use the Adam optimizer, it results in joint learning of a large fraction of both mislabeled and clean samples. Hence, offering a very low degree of separation. However, under the same training procedure, the SSFT still allows us to distinguish between the mislabeled and clean samples.

Stability across seeds and learning rates The pearson correlation for stability across seeds for the forgetting time metric is 0.83. This is higher than the corresponding learning time based metric (correlation 0.56). However, one of the drawbacks of our proposed metric is that the SSFT requires the use of an appropriate learning rate that allows the examples to be forgotten slowly. We provide more information about the same in the main paper.

Stability Across Learning Rate Schedules We experiment with three different learning rate schedules for the second-split training—



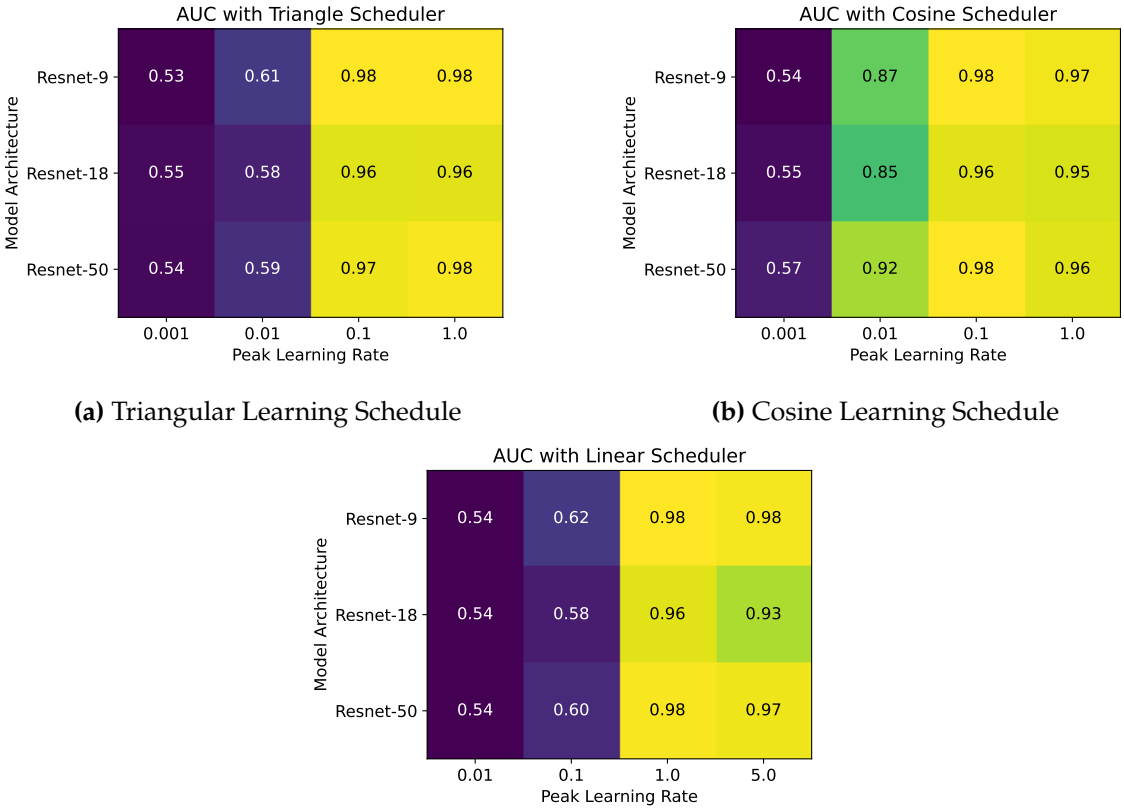
(a) **(b)**

Figure E.2: FSLT (First-split learning time) is able to provide some degree of separation between mislabeled and clean samples when trained with the SGD optimizer (left), but fails when the model is trained using Adam (right) on the CIFAR-10 dataset.

triangular, cosine and linear. In triangular learning rate, we increase the learning rate from 0 to the maximum set value linearly over the first 10 epochs, and then decay it back to 0 until we reach the last epoch (maximum of 100 epochs). In case of the linear schedule, we increase the learning rate from 0 to the maximum set value linearly over the course of 100 epochs. The intuition behind using a linear learning rate schedule was to be able to set higher peak learning rates so that the model eventually forget all the examples in the first split and we can create a better ordering between samples based on forgetting time (as opposed to the setting where only a small fraction of examples are ever forgotten).

The results of the combined analysis across hyperparameters such as architecture, learning rate and learning rate schedule are presented in the heatmap of AUC of mislabeled example detection in Figure E.3. The experiment was performed on the CIFAR-10 dataset with 10% label noise, and the forgetting times were averaged over 5 seeds before using them for AUC calculation. We can see that uniformly across architectures and learning rate schedules having a very low learning rate makes nearly all examples indistinguishable based on forgetting time. This is because all the models are sufficiently overparametrized to memorize all the examples in the dataset. Hence, when using a very small learning rate the optimization step moves the model weights insignificantly and we do not forget many mislabeled samples from the first split. On the other side, having a large learning rate helps achieve a strong separation between mislabeled and clean examples which also shows up in the form of high AUC values in the figure.

Impact of Sampling Frequency of Mislabeled Examples In the synthetic experiment performed in Section 10.4.2, we assumed that mislabeled examples occur from the majority subgroups. As a result, we observed that they get forgotten quickly dur-



(a) Triangular Learning Schedule

(b) Cosine Learning Schedule

(c) Linear Learning Schedule

Figure E.3: We present a heatmap for the AUC of detection of mislabeled examples using the SSFT metric under various learning rates, architecture sizes and learning rate schedules for the SGD optimizer. The experiment was performed on the CIFAR-10 dataset with 10% label noise, and the forgetting times were averaged over 5 seeds before using them for AUC calculation.

ing second-split training. However, in this section we aim to understand the impact of sampling frequency on the forgetting time of mislabeled examples. More specifically, we now assume that mislabeled samples occur in rare subgroups in the synthetic setup. We find that the learning curves of the mislabeled example stays the same as before, but the forgetting time for mislabeled examples closely approaches that of rare examples. This is because there is very little signal for the model to learn the opposite class during second split training since the example occurs only O(1) times. The learning and forgetting curves pertaining to the same experiment are presented in Figure E.4. In contrast with the forgetting curve in Section 10.4.2 where the mislabeled examples are quickly forgotten and their prediction is flipped, we find that when the subgroup corresponding to the mislabeled examples is infrequent, their forgetting time closely corresponds to that of rare examples; and on aggregate their predictions do not get flipped in the epochs that the model was trained for.

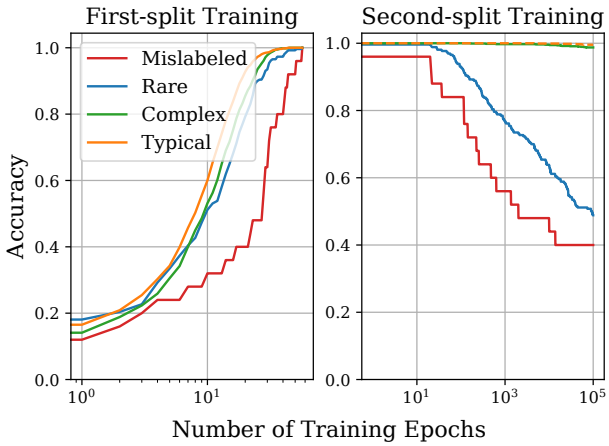


Figure E.4: Learning and forgetting curves for mislabeled and rare examples when the mislabeled examples are drawn from rare subgroups in the synthetic setup described in Section 10.4.2.

Mislabeled Example Detection In this section we provide additional details about the experimental setting for the results presented in Table 10.2. In case of the CIFAR-100 dataset, we reduce the learning rate for second-split training by a factor of 10, and use batch size of 128. While all the other training procedures in this paper used a cyclic learning rate, for the case of CIFAR-100, we use warm-up based multi-step decay learning rate schedule.[2] The model used for training was ResNet-18, and 10% label noise was added. The training setting for EMNIST is identical to that of the MNIST dataset. We use the first 10 classes of the dataset to make it a 10-class classification problem.

[2]We follow the code in https://github.com/weiaicunzai/pytorch-cifar100

Chapter F

FADU

Appendix

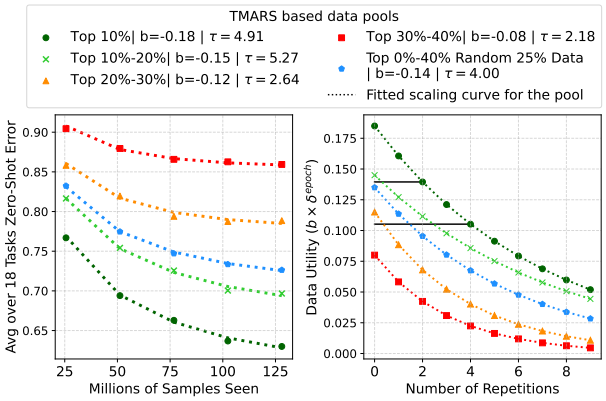


Figure F.1: Scaling curves with repeated data for visual-language models: We share the scaling curves for T-MARS based data pools, with average zeroshot performance across 18 tasks as the downstream performance metric.

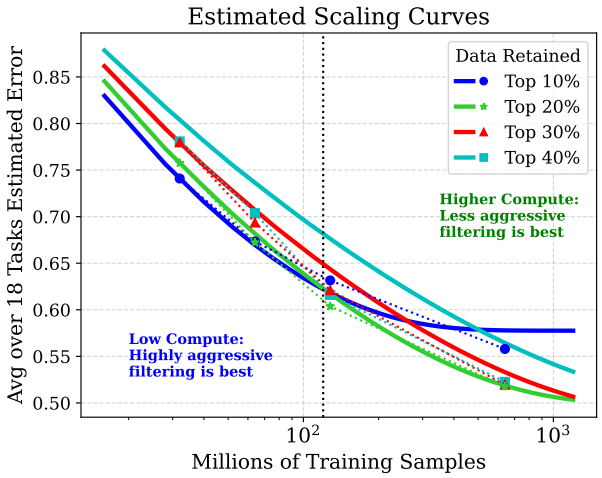


Figure F.2: Estimated Scaling curves for different data pool mixtures (data pools based on T-MARS scores). Again, we observe that highly aggressive filtering performs the best at low compute, while moderate filtering is better at higher computes.

Scaling curves for average performance across 18 tasks

In Figure 12.3 we showed the obtained scaling curves for each of the data pools, with ImageNet as the downstream performance metric. We share similar curves for each of the data pool, with average performance across 18 tasks as the downstream performance metric, in Figure F.1.

Finally, Figure F.2 shows the estimated scaling curves for different data pool mixtures, for average prformance across 18 tasks as the downstream performance metric.

Downstream Evaluation Datasets and Metrics

Following prior work [Radford et al., 2021a, Wortsman et al., 2021], we evaluate our models on a variety of image classification and retrieval datasets to assess their zero-shot capabilities. While the Datacomp [Gadre et al., 2023] benchmark averages performance across 38 different datasets, we use a subset of 18 such datasets where medium-scale models give better than random performance in order to be able to develop reliable scaling laws. More specifically, we select the following datasets:

1. ImageNet: a 1000-class image classi-

fication challenge [Russakovsky et al., 2015].

2. ImageNet-OOD: Six associated Imagenet distribution shifts—ImageNet-V2 [Recht et al., 2019], ImageNet-R [Hendrycks et al., 2020], ImageNet-A [Hendrycks et al., 2019], ImageNet-Sketch [Wang et al., 2019], ImageNet-O [Hendrycks et al., 2019], and ObjectNet [Barbu et al., 2019].

3. VTAB: 6 out of 12 datasets from the Visual Task Adaptation Benchmark [Zhai et al., 2020], including Caltech-101 [Fei-Fei et al., 2004], CIFAR10 [Krizhevsky, 2009], CIFAR100 [Krizhevsky, 2009], Oxford Flowers-102 [Nilsback and Zisserman, 2008], Oxford-IIIT Pets [Parkhi et al., 2012], and RESISC45 [Cheng et al., 2017].

4. Additional classification datasets: Food-101 Bossard et al. [2014], Pascal VOC 2007 [Everingham et al.], and Stanford Cars [Krause et al., 2013].

5. Retrieval: 2 retrieval tasks of MSCOCO [Chen et al., 2015] and Flickr [Young et al., 2014].

Most of the evaluation datasets constitute image-classification tasks. We use the 'Accuracy metric' to evaluate the zero-shot performance of the model on these datasets. The only exceptions include:

1. VTAB: We report 'Mean per Class Recall' for Caltech-101 [Fei-Fei et al., 2004], Oxford Flowers-102 [Nilsback and Zisserman, 2008], Oxford-IIIT Pets [Parkhi et al., 2012] datasets. This follows the standard evaluation protocol in past benchmarks [Gadre et al., 2023] and is done because of the large number of classes in these datasets.

2. Retrieval: For all the retrieval datasets we report the 'Mean Recall @ 1' which tells how probable is it for the top-recall entry to be relevant.

Scaling curves with CLIP Score as the Data Quality Metric

We share the scaling curves for data pools based on the CLIP score in Figure F.3. Similar to the trends observed in Sec. 12.3, we note that the magnitude of data quality parameter (b) decreases as we go towards worse pools. Finally, in Figure F.4 we show the scaling curves for combination of pools of varying quality, created using CLIP score filtering.

Pareto data filtering for T-MARS

Figure F.5 shows the variation in optimal data filtering strategy for T-MARS as the compute is scaled from 32M to 640M.

Finding the Optimal Scaling Parameters

Optimizing scaling parameters is a critical step in the validation of any scaling laws, particularly when these models are subjected to scaling laws that govern their behavior across different magnitudes of computation.

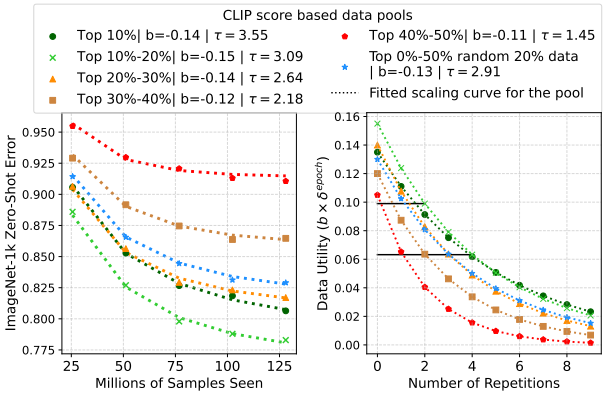


Figure F.3: Scaling curves for data pools based on CLIP score: We present the scaling curves for different data quality pools created using CLIP scores. Similar to observations in Figure 12.3, we note that the data quality parameter b decreases in magnitude as we go towards the worse bucket.

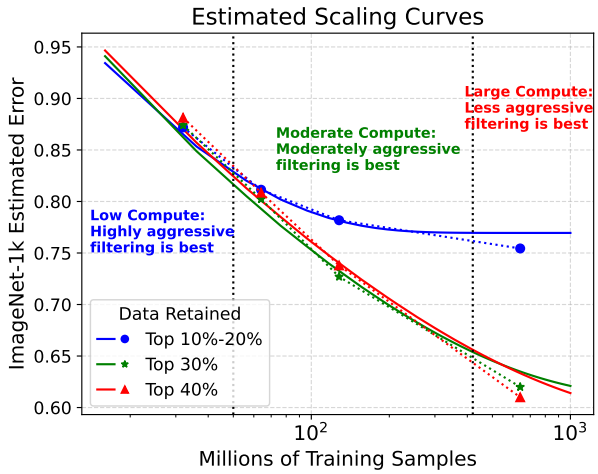


Figure F.4: Estimating scaling curve for combination of various data quality pools based on CLIP Score: We estimate the scaling curves for combination of data pools of various qualities, by modeling their mutual interaction (Theorem 1). Note that we do not train on data combinations to fit the above scaling curves (scatter points are test points), rather the scaling curves are estimated from the scaling parameters of individual pools.

This process, however, is fraught with challenges, primarily due to the sensitive nature of the optimization landscape, which is char-

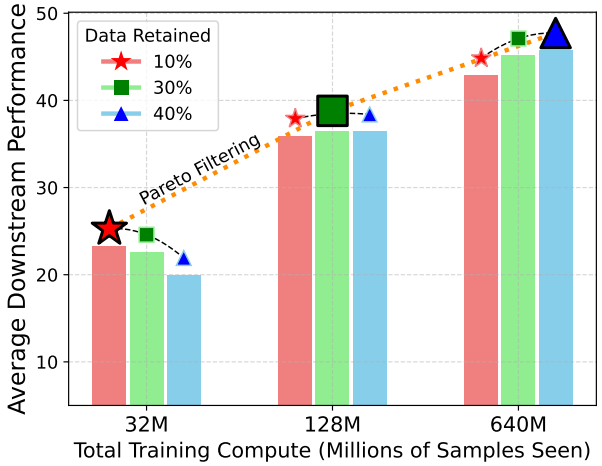


Figure F.5: We modify the state of the art data curation approach by changing the filtering threshold after ranking the data by their metric. While the original paper proposed retaining 30% of the data, our results show that depending on the ratio of compute to data pool size, we must adaptively make the filtering less (or more) aggressive to account for the diminishing utility of good data with repetitions. Results are presented on an average of 18 visual understanding tasks with a global data pool size of 128M samples, and varying compute scales.

acterized by numerous local minima and a complex objective function.

Challenges in Parameter Optimization In our initial approach to optimizing for the scaling parameters like a, b, c, τ we employed optimization libraries such as SciPy to find the best fit on the training points. Despite their widespread use, these methods often result in instability and inefficiency in the fitting process, especially being sensitive to initialization. The primary reason for this instability is the high sensitivity of the optimization problem, where slight variations in parameters can lead to significantly different outcomes, and at the same time, the existence of many solutions that fit the curve very well. As a remedy to this, we developed optimization methods ranging from gradient-based optimization to custom-built optimizers. In all of these approaches, we found that we could attain solutions that (a) had very high loss on the data points being optimized over; and (b) conversely, had a low loss on the points it was trying to minimize, but high loss on the points being extrapolated (mixture of data buckets). This was a rather unsatisfactory, because we should find scaling parameters *without* peeking at the points to extrapolate to.

Grid Search as a Stable Solution Given the limitations of conventional optimization methods, our work advocates for the use of grid search as a more stable and reliable method for determining optimal scaling parameters. Grid search, unlike heuristic or gradient-based optimization methods, systematically explores a specified parameter space, evaluating the performance of each parameter combination to identify the one that yields the best results. This exhaustive search process is especially advantageous in the context of scaling parameter optimization, where the landscape is sensitive to initialization. This approach mitigates the risk of overlooking optimal parameters due to the presence of local minima or the complexities of the objective function. Furthermore, grid search facilitates a reproducible parameter fitting process, enabling a clearer understanding of how different parameters influence the model's performance.

A view of the loss grid We perform a gridsearch for the normalizer $a \in [0.001, 1]$, data utility parameter $b \in [-0.005, -0.5]$, repetition parameter $\tau \in [1, 50]$ and $d \in [0.01, 0.02, 0.05, 0.10, 0.2]$. These grids were converged upon by continuously expanding the grid limits till the estimated optimal parameters for all the data pools lied in between the grid.

Finding normalizing constant a Recall from Sec. 12.2.2 that a refers to the normalizer term in the scaling equation which is kept constant across the pools. We learn a by performing a gridsearch over 100 possible values of $a \in [0, 1]$, and choosing the one which minimizes the combined loss across all the data quality pools. While we fix a to be same

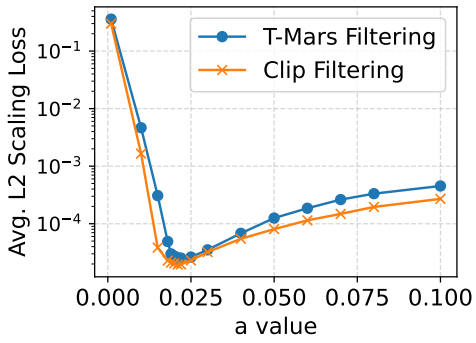


Figure F.6: Loss surface as we jointly optimize for a over the data pools. There exists a well-defined minima over our chosen grid-search range for a.

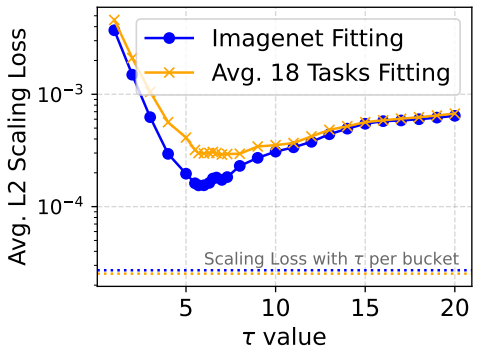


Figure F.7: Keeping the data diversity parameter (τ) different across the pools is necessary for good scaling curve fitting. Keeping the τ value same across the pools give a much worse (higher) L2 fitting loss compared to using different for each pool (see the horizontal dotted line).

across the pools, we let b and τ to vary. Figure F.6 shows the loss surface as we jointly optimize for a over data pools. We observe a well behaved loss surface over the gridsearch range, giving an optimal a value.

Learning the data repetition parameter τ Recall from Sec. 12.3 that we learnt different data repetition parameter (which denotes data diversity) τ for each of the data pools. One could alternatively chose to have the same diversity parameter across the pools, assuming that diversity varies uniformly in the web data. However, as shown in Figure F.7, learning a same τ value for all the buckets gives a much worse (high) L2 fitting error compared to learning different values for each of the pool (see the horizontal dotted curve).

What is the Axiom of Scaling?

Scaling laws have been studied for a while, especially when considering the scaling of large language models with increasing amounts of data [Kaplan et al., 2020, Hoffmann et al., 2022]. Since these formulations have abstracted away the 'per-sample' nature of scaling to the general formulation of the form $y = n^b$, it remains an open question to determine what is the actual 'axiom' of scaling. The answer to this question becomes extremely important in enabling us to study the dynamics of a mixture of data buckets of differing utilities, and that too, in an environment where the utility of data is constantly diminishing. In this section, we sequentially consider three different 'axioms' of scaling. We consider three different formulations that can potentially explain the sample level dynamics of scaling laws—(i) first, we assume that the derivative of the original scaling laws hold universally true, in our utility based formulation; (ii) second we consider the formulation that considers the effective data in the system at any point but considers the utility of data to be constant, as in Muennighoff

et al. [2023]; (iii) and finally a formulation that models the decay of data and utility simultaneously. In each of the discussions, we begin with certain assumptions, and examine if those assumptions hold at the event of a mixture of data buckets, that is, starting from an axiom, can we get back to that to predict the system loss in a mixture of data buckets.

Utility Based Decay We begin with our first scaling law formulation that models only the instantaneous system state and not the global state. The intuition behind such a setup is that it allows the mixing of different buckets naturally by calculating the instantaneous change in the loss by training on any sample from any data bucket. Assume that F.1 represents the axiom of scaling of loss with data seen.

$$\frac{dy}{dn} = \frac{y(n)}{n} b \tag{F.1}$$

The above can be thought of as the derivative of the more generally recognized scaling law $y = n^b$. Then, re-arranging we have that,

$$\frac{dy}{y} = \frac{dn}{n} b \tag{F.2}$$

$\Rightarrow$ Assume that 'b' utility term is a constant for a given region of the training curve. Then, between two such points in the state where we have seen n_1, n_2 samples respectively,

$$\int_{y_1}^{y_2} \frac{dy}{y} = \int_{n_1}^{n_2} \frac{dn}{n} b$$
$$\log \frac{y_2}{y_1} = b \log n_2 - b \log n_1$$

$$\log \frac{y_2}{y_1} = b \log \frac{n_2}{n_1} \tag{F.3}$$

Given a single data bucket, the assumption that 'b' is constant, should for instance, hold true during a given epoch. This gives us the following closed form relationship between loss values after seeing n_1, n_2 samples respectively.

$$y_2 = y_1 \left(\frac{n_2}{n_1} \right)^b \tag{F.4}$$

The closed-form solution for the rate of decay or loss, y_k, between training epochs is given by:

$$y_k = y_1 \left(\frac{n_2}{n_1} \right)^{b_2} \left(\frac{n_3}{n_2} \right)^{b_3} \cdots \left(\frac{n_k}{n_{k-1}} \right)^{b_k} \tag{F.5}$$

Where y_1 is the loss at the end of the first epoch, and

$$y_k = n_1^{b_1} \prod_{j=2}^{k} \left(\frac{n_j}{n_{j-1}} \right)^{b_j}.$$

The scaling term b is assumed to be a constant for a given region of the training curve. For example, during a given epoch, the relationship between y_2 and y_1 is given by $y_2 = y_1 \left(\frac{n_2}{n_1} \right)^b$.

Decay of the utility parameter Now, let us move into the paradigm of repeated epochs. We ask the question, how do we model the change in utility of a bucket as we see the dame data point multiple times. We assume that the utility parameter b decays exponentially at every epoch at a constant factor δ. This means that,

$$b^{(k)} = b^{(1)}\delta^{k-1}.$$

Estimating Data Mixtures under Utility-based decay Now that we have established the basic laws for the system state under the scaling laws modeling utility-based decay, let us examine how these assumptions play together in a setup of a mixture of buckets. For simplicity, whenever we consider a system with multiple data buckets, we will assume that the data points from these buckets are cycled alternately. Let us assume that we are at a given system state where the model loss and samples seen are determined by (y_0, n_0) respectively. Assume two data buckets parametrized by (b_1, δ_1) and (b_2, δ_2) as their respective initial utility, and utility decay rates. Let us first sample a data point from bucket 1, and then from bucket 2. Assume that the loss after seeing a single sample from the two buckets is y_1, y_2 respectively. Using the relationship in Equation F.4:

$$\frac{y_1}{y_0} = \left(\frac{n_0+1}{n_0}\right)^{b_1} = \left(1+\frac{1}{n_0}\right)^{b_1}$$
$$\frac{y_2}{y_1} = \left(\frac{n_0+2}{n_0+1}\right)^{b_2} = \left(1+\frac{1}{n_0+1}\right)^{b_2}$$

Using Taylor expansion and ignoring higher order term assuming that $n_0 \gg 1$, we can write the above as:

$$\begin{aligned}\frac{y_2}{y_0} &= \left(1+\frac{1}{n_0}\right)^{b_1}\left(1+\frac{1}{n_0+1}\right)^{b_2} \\ &\approx \left(1+\frac{1}{n_0}\right)^{b_1}\left(1+\frac{1}{n_0}\right)^{b_2} \\ &= \left(1+\frac{1}{n_0}\right)^{b_1+b_2}\end{aligned}$$

Let us assume that the effective utility of the mixture of data buckets is b_{eff}. Then, in the same steps between y_0 and y_2, we can write:

$$\frac{y_2}{y_0} = \left(1+\frac{2}{n_0}\right)^{b_{\text{eff}}}$$

Since the two formulations should be equivalent, we have that

$$b_{\text{eff}} = \frac{b_1+b_2}{2}$$

This gives us a way to estimate the effective utility of the mixture of data buckets at any given epoch k of training as,

$$b_{\text{eff}}^{(k)} = \frac{b_1\delta_1^{k-1} + b_2\delta_2^{k-1}}{2}$$

As we move into the next formulation, let

us highlight that the key challenge behind the validity of this formulation is the initial axiom we assumed. We assume that the system is determined by instantaneous interactions where the effective number of samples seen is constant, independent of the number of times it has been seen before, and only its utility decreases every time.

Effective Data Based Formulation Let us now consider a formulation that only accounts for the 'effective data' in the system to predict the loss of the model on seeing 'n' samples. The central assumption behind this formulation is that the 'utility' of data itself does not decay, and stays constant throughout the training process, rather only the effective data in the system decays. This is the formulation that has been considered in the past by Muennighoff et al. [2023]. Let us consider that the 'axiom' of scaling law is now represented by the following equation:

$$\begin{aligned} y &= n_{\text{eff}}^{b}, \\ n_{\text{eff}} &= \eta(1 + \delta + \delta^2 + \cdots + \delta^{k-2}) + \gamma\delta^{k-1} \end{aligned} \tag{F.6}$$

where η represents the number of unique samples in the training data, δ represents the rate of decay of effective data when seeing a repeated epoch of the data, and $\gamma < \eta$ represents the number of examples seen in the current epoch of training. This suggests that at a given system state, $dn_{\text{eff}} = \delta^{k-1}dn$

$$\begin{aligned} \frac{dy}{dn_{\text{eff}}} &= b\frac{y(n_{\text{eff}})}{n_{\text{eff}}} \\ \frac{dn_{\text{eff}}}{dn} &= \delta^{k-1} \end{aligned}$$

Note that since the system state has a fixed utility value throughout training, we can use the formulation from Equation F.4 in terms of n_{eff} as follows:

$$y_2 = y_1\left(\frac{n_{\text{eff2}}}{n_{\text{eff1}}}\right)^{b}$$

Estimating Data Mixtures under Effective Data-based decay Let us now consider the case where we have a mixture of data buckets, and we are trying to estimate loss of the system characterized by the mixture of data buckets. As before, let us assume that we are at a given system state where the model loss and samples seen are determined by (y_0, n_0) respectively. Assume two data buckets parametrized by (η_1, δ_1) and (η_2, δ_2) as their respective initial effective data, and effective data decay rates. Let us first sample a data point from bucket 1, and then from bucket 2. For simplicity, we will consider the

case of training on the second epoch of data:

$$\frac{y_1}{y_0} = \left(\frac{n_0 + \delta_1}{n_0}\right)^{b_1} = \left(1 + \frac{\delta_1}{n_0}\right)^{b_1}$$
$$\frac{y_2}{y_1} = \left(\frac{n_0 + \delta_1 + \delta_2}{n_0 + \delta_1}\right)^{b_2} = \left(1 + \frac{\delta_2}{n_0 + \delta_1}\right)^{b_2}$$
$$\frac{y_2}{y_0} = \left(1 + \frac{\delta_1}{n_0}\right)^{b_1}\left(1 + \frac{\delta_2}{n_0 + \delta_1}\right)^{b_2}$$
$$\approx \left(1 + \frac{\delta_1}{n_0}\right)^{b_1}\left(1 + \frac{\delta_2}{n_0}\right)^{b_2}$$
$$= \left(1 + \frac{\delta_1}{n_0}\right)^{b_1}\left(1 + \frac{\delta_2}{n_0}\right)^{b_2}$$

By Taylor expansion, and keeping only the first order terms, in the expansion of $(1+x)^a$ for very small x, we have that $(1+x)^a \approx 1+ax$. This gives us that,

$$\frac{y_2}{y_0} \approx \left(1 + \frac{\delta_1}{n_0}\right)^{b_1}\left(1 + \frac{\delta_2}{n_0}\right)^{b_2}$$
$$\approx \left(1 + \frac{\delta_1}{n_0}b_1\right)\left(1 + \frac{\delta_2}{n_0}b_2\right)$$
$$\approx 1 + \frac{\delta_1 b_1 + \delta_2 b_2}{n_0}$$

But, we know that this should be equivalent to the formulation where we consider a fixed utility of data throughout training, given by b_{eff}.

$$\frac{y_2}{y_0} = \left(1 + \frac{\delta_1 + \delta_2}{n_0}\right)^{b_{\text{eff}}}$$

Since the two formulations should be equivalent, we have that

$$b_{\text{eff}} = \frac{\delta_1 b_1 + \delta_2 b_2}{\delta_1 + \delta_2}$$

Notice that in the final formulation, b_{eff} is a weighted average of the utility of the data in the two buckets, with the weights being the effective data in the system at the time of training. However, this contradicts our initial assumption that it is only the data that decays in the system, and the utility stays constant. While the utility does not 'decay', it does get re-weighted as the effective data in the system changes. This is a key insight that we gain from this analysis, which requires us to consider the change in both the effective data and the utility of data in the system. Note that these results still do not violate the formulation proposed in the work of Muennighoff et al. [2023], because they consider data to come from a single bucket. This means that $b_{\text{eff}} = {}^{\delta_1 b_1}/_{\delta_1} = b_1$. However, in the case of a mixture of data buckets, the utility of data in the system is a weighted average of the utility of data in the individual buckets.

An alternate paradigm in which such an equation may work is when the rate of decay (δ) associated with all buckets is the same. However, we find that adding the constraint of fixed decay factor between all buckets leads to significantly higher loss even on the points to be fit for scaling law, let alone the extrapolation points.

Decaying Effective Data with changing Utility Based on the insights from the previous two formulations, we now consider a formulation that models the decay of both the utility of data and the effective data in the sys-

tem. It is clear from the previous part that we need a formulation where we can model the instantaneous utility of data in order to be able to mix different data buckets. The key difference from the formulation based on only utility decay is that we consider that there is a decay factor associated with the effective number of samples in the system. Let us first rewrite the instantaneous system state in terms of the effective data and utility of data as follows:

$$\begin{aligned} \frac{dy}{dn_{\text{eff}}} &= \frac{y(n_{\text{eff}})}{n_{\text{eff}}} b, \\ \frac{dn_{\text{eff}}}{dn} &= \delta^{k-1} \end{aligned} \tag{F.7}$$

From the formulation in the previous part in Section F, it then follows that,

$$b_{\text{eff}}^{(k)} = \frac{b_1 \delta_1^{k-1} + b_2 \delta_2^{k-1}}{\delta_1^{k-1} + \delta_2^{k-1}},$$

where $b_{\text{eff}}^{(k)}$ is the effective utility of the mixture of data buckets at any given epoch k of training.

Using the above, we can consider the closed form solution for the loss of the system at any given epoch k of training as a product of the loss at the end of the first epoch, and the ratio of the effective data seen at the end of the kth epoch to the power effective utility.

$$y_k = y_1 \left(\frac{n_{\text{eff2}}}{n_{\text{eff1}}}\right)^{b_{\text{eff}}^{(2)}} \left(\frac{n_{\text{eff3}}}{n_{\text{eff2}}}\right)^{b_{\text{eff}}^{(3)}} \cdots \left(\frac{n_{\text{effk}}}{n_{\text{eff(k-1)}}}\right)^{b_{\text{eff}}^{(k)}}$$

$$y_k = y_1 \prod_{j=2}^{k} \left(\frac{n_{\text{effj}}}{n_{\text{eff(j-1)}}}\right)^{b_{\text{eff}}^{(j)}}$$

Estimating Data Mixtures under Decaying Effective Data with changing Utility We will directly use the data mixing results from F to estimate the loss of the system at any given epoch k of training. Note that this is governed under the assumptions of exponential decay of data with repetitions.

Equivalence of Formulations F.1 and F.3 We will now show that under certain conditions, the decay of data and utility, and the decay of just utility reduce to the same formulation. Let us consider the case where the decay factor of the effective data is the same as the decay factor of the utility of data. Consider two buckets parametrized by (b_1, δ) and (b_2, δ) as their respective utility and decay factor. We will now study the ratio of the two losses after seeing a single sample from the two buckets. Once again assume that the initial system state is characterized by (y_0, n_0). For simplicity, we will consider the case of training on the second epoch of data:

Effecttive Utility Formulation Recall that

$$b_{\text{eff}} = \frac{b_1 \delta^{k-1} + b_2 \delta^{k-1}}{2}.$$

Then, we can write the loss after seeing a single sample as:

$$\begin{aligned}\frac{y_1}{y_0} &= \left(1+\frac{1}{n_0}\right)^{b_{\text{eff}}} \\ &= \left(1+\frac{1}{n_0}\right)^{\frac{b_1\delta_1+b_2\delta_2}{2}} \\ &\approx \left(1+\frac{1}{n_0}\right)^{\frac{b_1\delta_1+b_2\delta_2}{2}} \\ &\approx \left(1+\frac{b_1\delta_1+b_2\delta_2}{2n_0}\right)\end{aligned}$$

Effective Data, Dynamic Utility Based Formulation In this case, the

$$\begin{aligned}b_{\text{eff}} &= \frac{\delta_1 b_1+\delta_2 b_2}{\delta_1+\delta_2} \\ \delta_{\text{eff}} &= \frac{\delta_1+\delta_2}{2}\end{aligned}$$

Then, we can write the loss after seeing a single sample from the two buckets as:

$$\begin{aligned}\frac{y_1}{y_0} &= \left(1+\frac{\delta_{\text{eff}}}{n_0}\right)^{b_{\text{eff}}} \\ &= \left(1+\frac{\delta_1+\delta_2}{2n_0}\right)^{\frac{\delta_1 b_1+\delta_2 b_2}{\delta_1+\delta_2}} \\ &\approx \left(1+\frac{\delta_1 b_1+\delta_2 b_2}{2n_0}\right)\end{aligned}$$

Notice that, under the Taylor approximation, and assumption that $n_0 \gg 1$, the two formulations are equivalent. This suggests that under the assumption that the decay factor of the effective data is the same as the decay factor of the utility of data, the two formulations reduce to the same formulation. This is a key insight that we gain from this analysis, which requires us to consider the change in both the effective data and the utility of data in the system. Note that these results still do not violate the formulation proposed in the work of Muennighoff et al. [2023], because they consider data to come from a single bucket. This means that $b_{\text{eff}} = {}^{\delta_1 b_1}/_{\delta_1} = b_1$. However, in the case of a mixture of data buckets, the utility of data in the system is a weighted average of the utility of data in the individual buckets.

Rate of Repetition Decay

Conventionally scaling laws have been studied in the context of language models [Kaplan et al., 2020, Hoffmann et al., 2022]. To the best of our knowledge, the only large scale attempt at predicting scaling of vision language models was done in the work of Cherti et al. [2023]. However, the extrapolated graphs in their work showed large errors as compared to the actual performance of the models. Hence, the scaling of vision language models is still an open question, especially when trained under contrastive loss.

Challenge: Squared Contrastive Pairs This is a fundamentally new challenge because in a data pool of N samples, there exist N^2 pairs of comparisons that contribute to the loss. In such a setting, *should the utility of a data pool still diminish exponentially after seeing N samples, or should it diminish after seeing N^2 pairs?* This is a fundamental question that we address in this section.

The contrastive loss for any given batch of data can be given by the following equation:

$$\mathcal{L} = -\log \frac{\ell((x_i, t_i))}{\sum_{k=1}^{B-1} \ell(x_i, t_k)}, \tag{F.8}$$

where B is the batch size, and ℓ is some loss function based on the cosine similarity between the embeddings of the image and the text. x_i and t_i are the image and text embeddings of the ith sample in the batch. The goal is to maximize the similarity between the image and text embeddings of the same sample, and minimize the similarity between the embeddings of different samples. We can decompose the loss into the numerator and the denominator. In a given batch, there are a total of B samples seen in the numerator, and $B \times (B-1)$ samples seen in the denominator. We will call this $B \times B$ for simplicity.

Let us assume that the training set up uses a batch size of B for contrastive learning. Given a dataset of N samples, the number of batches seen in a single epoch is given by $\frac{N}{B}$. Therefore, the total number of comparisions seen in the denominator in a single epoch is given by $N \times B$.

Utility Decay Assumption We assume that the utility decays exponentially, however, only after seeing each unique comparison in the dataset. For a data pool of N samples, while conventionally such a decay event would happen every epoch after seeing N samples, in the case of contrastive learning, this decay event should happen after seeing N^2 pairs of samples. This means that the number of epochs after which the utility of the data pool decays should be given by $\frac{N^2}{N \times B} = \frac{N}{B}$.

Decay formulation Let us assume that δ_{g} is the gold rate of decay after seeing N^2 samples. Since the modeling till now was done based on utility decay per epoch, we can write a relationship between the gold rate of decay and the rate of decay after seeing N samples as follows:

$$\delta_g = \delta^{N/B} = \frac{1}{2}^{N/B\tau}$$

Finding decay value on merging data buckets When we merge data buckets, a unique phenomenon happens. The overall pool size increases. While in the conventional language modeling paradigm this may not have a significant impact, in the case of contrastive learning, this changes decay rate because the number of comparisons in the denominator increase at the squared rate of total samples. Using δ_g we can find the new value of $\hat{\tau}$ as follows:

$$\delta_g = \frac{1}{2}^{N/B\tau} = \frac{1}{2}^{\hat{N}/B\hat{\tau}}$$
$$\hat{\tau} = \frac{\hat{N}}{N}\tau$$

Hence, $\tau_p = p\tau$ when merging p different buckets of a given size.

Finding the best decay rate In the above discussion, we assumed that the rate of decay should happen after seeing N^2 pairs of sam-

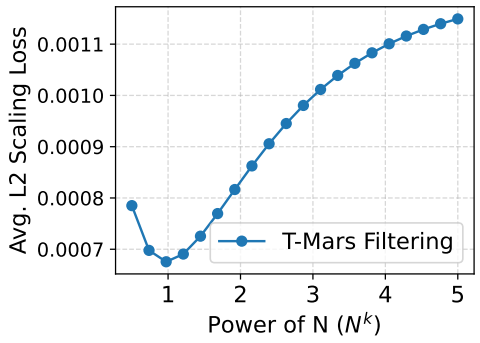


Figure F.8: Loss surface as we find the best rate of decay of utility as a function of N^k.

ples. However, this is an assumption that we made. To test how this empirically holds with respect to different rates of change in N we allow for the flexibility of $\tau_p = p^k\tau$. Then we run an estimation analysis, at various values of k as in Figure F.8 to find the best value of k that minimizes the loss on the data points to be fit. We notice that the results of the theoretically derived relationship exactly match the empirically determined value of minimum loss at $k = 1$. Hence, for the purposes of all the merging experiments, we use $\tau_p = p\tau$ as the rate of decay of utility of data in the system when merging p different data buckets.

Chapter G

Diffusion Model Hallucinations

Additional Experimental Details

Gaussian experiments We run all our experiments for 10,000 epochs with batch size of 10,000. A linear noise schedule is used with starting noise $\beta_0 = 0.001$ and the final noise $\beta_1 = 0.2$. We use $T = 1000$ by default in our experiments (unless specified otherwise). The neural network (NN) is trained to predict the noise (similar to the original DDPM [Ho et al., 2020] implementation) and we use a Mean Squared Error loss to train the model. The input and output of the NN have the same shape (in this case, 1 for 1D Gaussian and 2 for the 2D Gaussian). The NN architecture starts with an initial fully connected layer, followed by three blocks and then output fully connected layer. Each block includes normalization, a LeakyReLU activation, and two fully connected layers. Finally, the output is normalized and transformed back to the input dimension with a fully connected layer. Adam [Kingma and Ba, 2014] with learning rate of 0.001 is used as the optimizer. We build our codebase on top of [1] for the synthetic toy experiments.

Metric: We use $t = 0$ to $t = 15$ (last 15 steps in the reverse diffusion process) to compute the variance of the trajectory in the case of Gaussian 1D and $t = 0$ to $t = 8$ in the case of 2D Gaussian Grid.

Shapes The generated images are grayscale images of size 64×64. A total of 5000 images is generated for training the diffusion model. We use a U-Net [Ronneberger et al., 2015] architecture to model the reverse diffusion process. We use a cosine noise scheduler similar to ADM [Nichol and Dhariwal, 2021]. We derive our implementation based on [2] for training the DDPM. We train an unconditonal DDPM on the dataset with $T = 1000$ while training and 250 steps during sampling to reduce computation cost [Song et al., 2021].

MNIST MNIST [LeCun et al., 1998] consists of 60,000 grayscale images of size (28, 28). We use classifier-free guidance [Ho and Salimans, 2022] to train a conditional DDPM on MNIST with $T = 500$. For each generation, we train for a total of 50 epochs with a batch size of 512 shared across 4 GPUs. Adam [Kingma and Ba, 2014] optimizer with learning rate of 1e-4 is used to train the network. We use a U-Net [Ronneberger et al., 2015] with 256 feature dimension to model the reverse diffusion process. For the variance filtering mechanism in Section 15.4, we use 10 timesteps between $t = 100$ to $t = 150$

[1] https://github.com/tqch/ddpm-torch

[2] https://github.com/VSehwag/minimal-diffusion

to compute the variance of the trajectory. In the case of MNIST, we do post-hoc filtering just using the samples. This means that we add t timesteps of noise, then compute $\hat{x_0}$ and then use this to compute variance.

Our implementations of the DDPM model is based on PyTorch [Paszke et al., 2019].

Compute: We run all our experiments of Nvidia RTX 2080 Ti and Nvidia A6000 GPUs. The training and sampling for the Gaussian experiments takes less than 3 hours on single 2080Ti GPU. Sampling 100 million datapoints takes around 3-4 hours. Running DDPM on the shapes dataset takes around 6-7 hours with 4 2080Ti GPUs. The recursive generative training on MNIST takes about 16 hours with 4 A6000 GPUs for 5 generations.

Hands We trained for a total of 200k iterations with batch size 16 and a learning rate of 1e-4. The diffusion process was trained with 1000 timesteps ($T = 1000$) with a cosine noise schedule. The U-Net comprised 256 channels, with an attention mechanism incorporating 64 channels per head and 3 residual blocks. For sampling, we use 500 timesteps with respacing. We base our implementation and hyperparameters on the official DDPM-IP [Ning et al., 2023] repository.

Limitations and Broader Impact

Hallucinations in LLMs have been studied extensively [Zhang et al., 2023c, Ye et al., 2023] given the widespread use of these systems in various contexts. This work investigates hallucinations in diffusion models. In current generative models, these hallucinations could be used to more easily identify machine-generated images. Developing a metric to identify these hallucinations and remove them could make the detection of generated images much harder. However, we argue that understanding hallucinations in diffusion models is crucial as it can help shed light on their failure modes and thereby enable better control in practical applications.

In current text-to-image generative models, the poorly modeled "hands" are a clear giveaway in identification of AI generated images. The detection of such AI-generated content would be made much more difficult if these hallucinations were identified and removed from the generated images. While our work builds an understanding of hallucinations, and allows us to also detect them, we believe that future generations of models would have become more robust to such hallucinations by virtue of training on more data independent of this work.

Concerning the limitations of the proposed hallucination metric, the selection of the right timesteps is key to be able to detect hallucinations. More analysis on what region of trajectory leads to hallucinations would be useful across various schedules and sampling algorithms. We believe these are great areas for future work to explore. Additional explorations of mode interpolation and hallucinations in real-world datasets would be useful to the community.

Additional Experiments and Figures

We also study Variational Diffusion Models (VDM) [Kingma et al., 2021] to verify the generality of our findings. Our results show that the over-smoothed score function phenomenon persists in VDM, supporting the hypothesis that this issue is not specific to DDPM. We train a simple VDM on the 2D Gaussian with 10k samples. We follow the setup and hyperparameters in the official implementation [3]. We train both continuous and discrete variants of VDM on the 2D Gaussian dataset. The main observation is that VDM mitigates the hallucinations significantly especially with more training data but the phenomenon of mode interpolation still exists. In this figure, we also show the impact of the number of sampling steps on the count of hallucinations. We clearly see that increasing the number of sampling steps reduces the number of hallucinated samples. This can be clearly observed in Figure G.2 (first two columns) where the count of hallucinations decreases mode interpolation.

The frequency of mode interpolation is inversely proportional to the number of training samples. We train the unconditional diffusion model with 25k, 50k, 100k and 500k samples from the true distribution.

1. Figure G.3 shows the histogram of samples generated by the diffusion model (with 10 million samples) when the model is trained on the distribution with $\mu_1 = 1, \mu_2 = 2, \mu_3 = 3$.
2. Figure G.4 shows the histogram of samples generated by the diffusion model (with 10 million samples) when the model is trained on the distribution with $\mu_1 = 1, \mu_2 = 2, \mu_3 = 4$.
3. We also experiment with mixture of 2 Gaussians in Figure G.5 and 4 Gaussians in Figure G.6.
4. Figure G.7 shows the FID, precision and recall curves for MNIST across generations.
5. Figure G.8 shows additional examples of hallucinated images generated by the diffusion model.
6. Figure G.9 shows the $\hat{x_0}$ across various timesteps for a hallucinated image. The number on top of the image indicates the timestep.
7. Figure G.10 shows the $\hat{x_0}$ across various timesteps for a image in-support of the distribution. The number on top of the image indicates the timestep.

[3] https://github.com/google-research/vdm

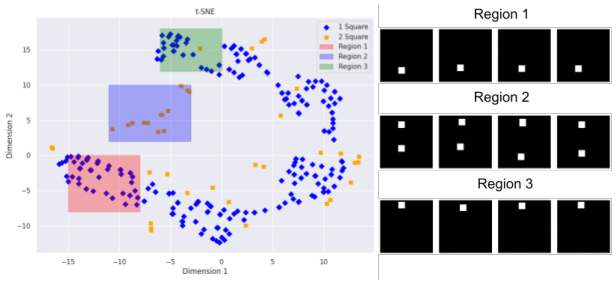


Figure G.1: Interpolation in Representation Space. We analyze the bottleneck of the U-Net to demonstrate mode interpolation in the Shapes dataset. We clearly see that Region 2 (which consists of 2 squares) is interpolating between Region 1 (one square in the bottom half) and Region 3 (one square in the top half).

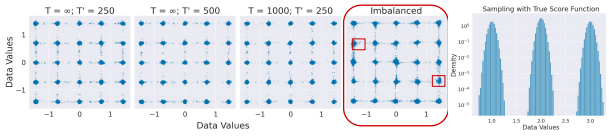


Figure G.2: Variational Diffusion Model. We train a Variational Diffusion Model (VDM) on the 2D Gaussian Data with 10k samples (first three columns). T denotes the timesteps during training and T' denotes the sampling timesteps. $T = \infty$ refers to the continuous time variant. The fourth column shows a DDPM trained on a 2D Gaussian with imbalanced modes. The boxes indicate the modes with less data. The last column shows result of sampling from the true score function.

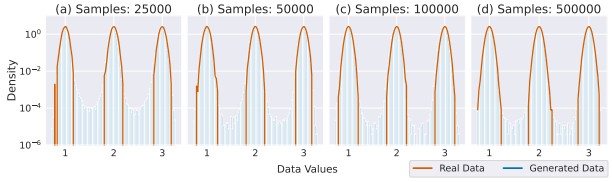


Figure G.3: Mixture of 3 Gaussians with $\mu = [1, 2, 3]$. We vary the number of training samples and observe that mode interpolation decreases with increase in the size of training data

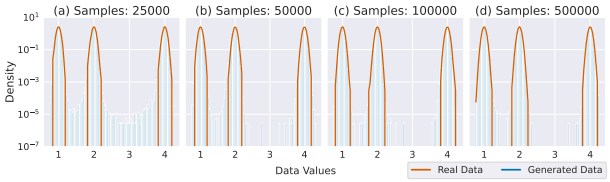


Figure G.4: Mixture of 3 Gaussians with $\mu = [1, 2, 4]$. We vary the number of training samples and observe that mode interpolation decreases with increase in the size of training data.

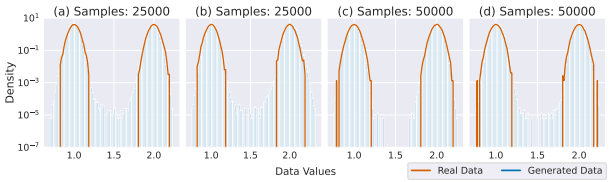


Figure G.5: Mixture of 2 1D Gaussians with varying number of training samples. (a) and (b) have the same number of training samples but with two different seeds. Similarly for (c) and (d).

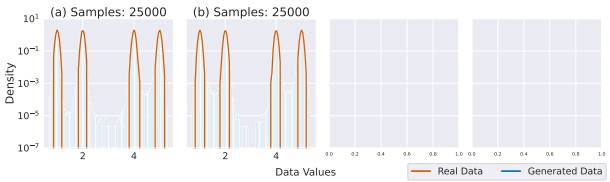


Figure G.6: Mixture of 4 1D Gaussians ($\mu = [1, 2, 4, 5]$) with varying number of training samples. (a) and (b) have the same number of training samples but with two different seeds. We clearly see more samples in the region between modes $\mu_1 = 1$ and $\mu_2 = 2$ compared to $\mu_2 = 2$ and $\mu_3 = 4$.

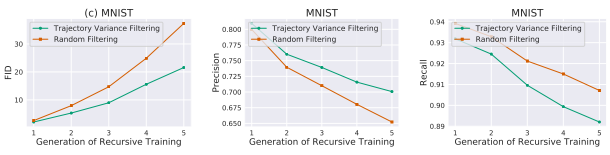


Figure G.7: Recursive Generative Training on MNIST with Variance and Random Filtering. We observe that the proposed filtering mechanism can discard low quality samples while maintaining sufficient diversity.

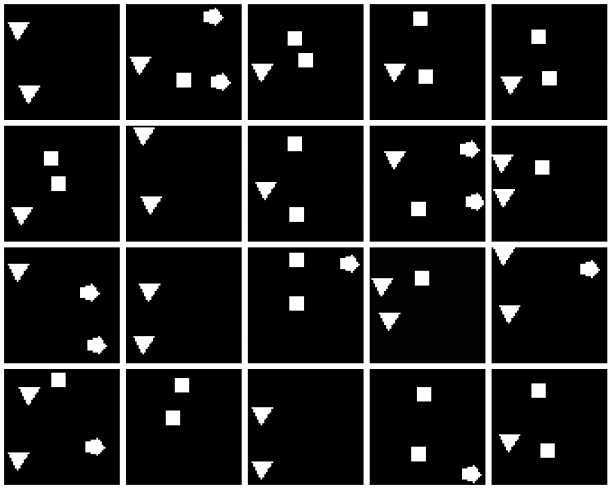


Figure G.8: Example of Generated Hallucinated Images

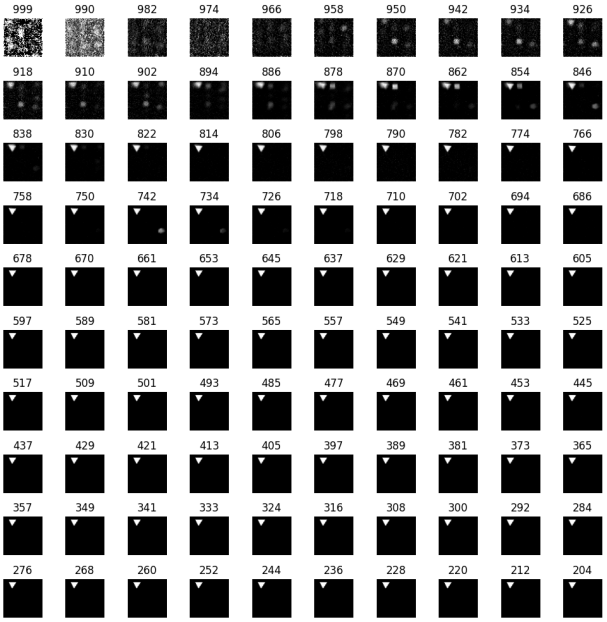


Figure G.10: $\hat{x_0}$ for In-Support Sample. Here, we observe that the predicted x_0 is more consistent around $t = 700$ to $t = 850$.

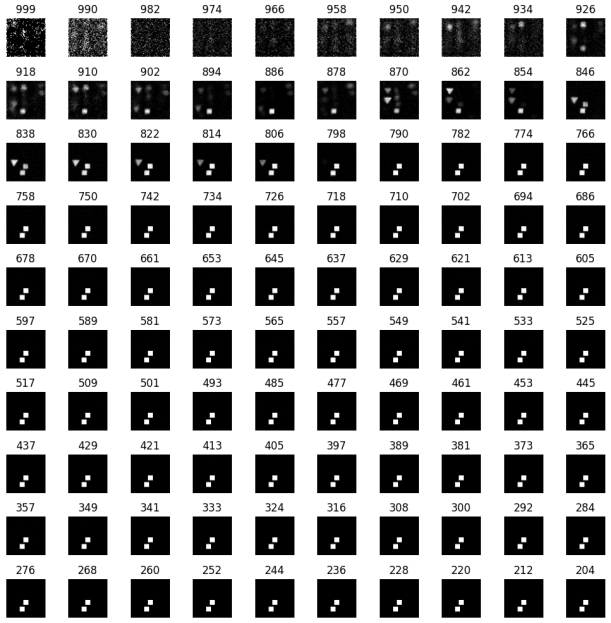


Figure G.9: $\hat{x_0}$ for Hallucinated Sample. Here, we observe that the predicted x_0 has a lot of variance around $t = 700$ to $t = 850$. This clearly motivates our proposed metric.

Figure G.11: Hallucinated Images of Hands generated by the diffusion model.

Figure G.12: In-Support Images of Hands generated by the diffusion model.

Figure G.13: $\hat{x_0}$ trajectory for Hallucinated Sample (with 250 timesteps). We observe high variance/instability during the steps $t = 600$ to $t = 900$.

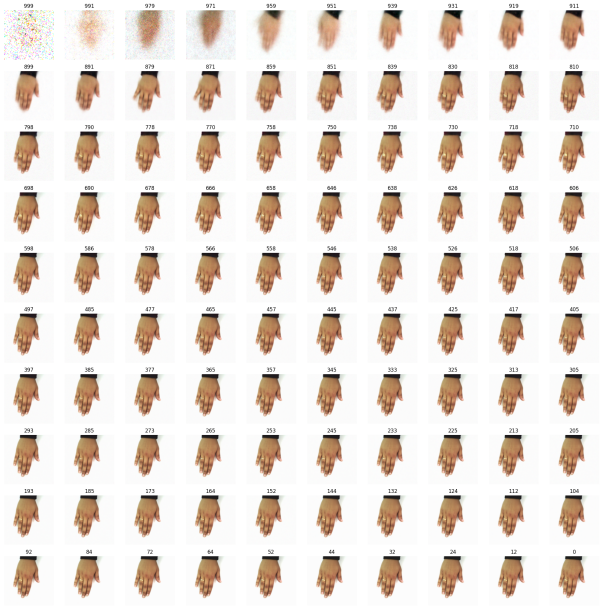


Figure G.14: $\hat{x_0}$ trajectory for In-Support Sample (with 250 timesteps). We do not observe high variance/instability during the steps $t = 600$ to $t = 900$.

Chapter H

Rephrasing the Web (WRAP)

Large language models are trained on massive scrapes of the web, which are often unstructured, noisy, and poorly phrased. Current scaling laws show that learning from such data requires an abundance of both compute and data, which grows with the size of the model being trained. This is infeasible both because of the large compute costs and duration associated with pre-training, and the impending scarcity of high-quality data on the web. In this work, we propose **W**eb **R**ephrase **A**ugmented **P**re-training (T-MARS) that uses an off-the-shelf instruction-tuned model prompted to paraphrase documents on the web in specific styles such as "like Wikipedia" or in "question-answer format" to jointly pre-train LLMs on real and synthetic rephrases. First, we show that using T-MARS on the C4 dataset, which is naturally noisy, speeds up pre-training by $\sim 3\times$. At the same pre-training compute budget, it improves perplexity by more than 50% on average across different subsets of the Pile, and improves zero-shot question answer accuracy across 13 tasks by more than 2%. Second, we investigate the impact of the re-phrasing style on the performance of the model, offering insights into how the composition of the training data can impact the perfor-

Dataset Details

Training Dataset The primary pretraining corpus in our experiments is Colossal Clean Crawled Corpus (C4), a curated English text dataset comprising over 170 billion tokens. This corpus is derived from CommonCrawl, a common practice in the pretraining of LLMs [Brown et al., 2020a, Raffel et al., 2020, Touvron et al., 2023]. This data source is also prominently featured in openly available LLM pretraining corpora, including The Pile [Gao et al., 2020] and RedPajama [Computer, 2023]. There are different versions of CommonCrawl data and our selection of C4 for pretraining is driven by driven by its size and quality.

We also compare with pre-training on the Refined Web corpus [Penedo et al., 2023]. The dataset is also derived from the CommonCrawl, however has a more stringent filtering process. Our selection of Refined Web is for comparing synthetic rephrases to high quality subsets of web data, which were shown to achieve similar performance compared with curated datasets [Penedo et al., 2023]. For our experiments we used the first 3050 files and train for 300B tokens to match training on C4. We aso conduct experiments with the first 1650 files to account for multiple epochs on the Refined Web dataset.

Pile Perplexity Evaluation For the evaluation phase, we employed 20 subsets from the Pile corpus. We excluded the Europarl subset because it contained non-English language. The subsets used are: CC, StackExchange, Wikipedia, GitHub, PubMed Abstracts, Openwebtext2, Freelaw, Math, NIH, USPTO, Hackernews, Enron, Books3, PubMed Central, Gutenberg, Arxiv, Bookcorpus2, Opensubtitles, Youtubesubtitles, Ubuntu, and Philpapers. We take the first 10000 samples from each subset and split into documents of maximum length 1024. The reported average in all perplexity plots is a weighted average over the perplexity of all domains according to the ratios in Table H.1.

Pile Weighted Average Ratios We report the ratios for samples according to the first 10,000 documents from our Pile validation set in Table H.1. Note that there are some slight variations in the ratios compared with those reported in [Gao et al., 2020], but most ratios are similar.

Zero-shot Evaluation Dataset We evaluate our models on a total of 13 different zero-shot benchmarks to assess their abilities across various natural language tasks. These benchmarks are categorized into two subsets: Specialized Knowledge and General Understanding.

Specialized Knowledge This subset comprises datasets that focus on domain-specific knowledge and expertise.

- **ARC Challenge (ARC-C)**: This dataset is part of the AI2 Reasoning Challenge (ARC) [Clark et al., 2018c], containing science exam questions from grades 3 to 9. The ARC Challenge set includes

Dataset	Our Ratio (%)	Ratio (%)
ArXiv	10.4	9.0
BookCorpus2	0.8	0.8
Books3	11.8	12.1
Pile-CC	14.0	18.11
Enron	0.1	0.1
EuroParl	1.1	0.7
FreeLaw	5.3	6.1
Github	10.9	7.6
Gutenberg	1.5	2.2
Hackernews	0.6	0.6
Dm Mathematics	2.0	1.2
NIH	0.2	0.3
OpenSubtitles	1.3	1.6
OpenWebText2	8.2	10.0
PhilPapers	0.7	0.4
PubMed Abstracts	0.7	3.1
PubMed Central	14.9	14.4
StackExchange	5.8	5.1
Ubuntu	1.3	0.9
USPTO	2.7	3.7
Wikipedia	3.4	1.5
YoutubeSubtitles	0.6	0.6

Table H.1: Pile ratios for our evaluation compared with published ratios.

more difficult questions that necessitate higher-order reasoning.

- **SciQ**: A dataset of science exam questions, specifically designed to evaluate the ability of NLP models in understanding and reasoning within the scientific domain [Johannes Welbl, 2017].
- **PubMedQA**: This dataset focuses on biomedical literature and is designed to evaluate the understanding of medical and healthcare-related information [Jin et al., 2019].
- **MathQA**: This dataset challenges models in mathematical problem-solving, requiring both numerical understanding and reasoning skills [Amini et al., 2019].
- **MMLU**: Multi-domain question answering, MMLU assesses the model's expertise over a wide range of specialized subjects, from professional domains to academia [Hendrycks et al., 2021d].

General Understanding This subset contains datasets that test general cognitive skills, language understanding, and common sense reasoning.

- **ARC Easy (ARC-E)**: The Easy set of the AI2 Reasoning Challenge [Clark et al., 2018c] features questions from the same source as ARC-C but are considered less challenging and do not require as advanced reasoning skills.

- **BoolQ**: A dataset consisting of boolean (yes/no) questions, focusing on reading comprehension and general understanding of natural language text [Clark et al., 2019].
- **Winogrande (Wino.)**: This dataset challenges models on common sense reasoning in a language context, focusing on pronoun disambiguation tasks [Sakaguchi et al., 2021].
- **PIQA**: Physical Interaction Question Answering tests the understanding of everyday physical processes, an aspect of practical common sense [Bisk et al., 2020a].
- **HellaSwag**: This dataset evaluates a model's ability to complete scenarios in a contextually and logically coherent manner, requiring both language understanding and common sense reasoning [Zellers et al., 2019].
- **TruthfulQA**: Focused on the generation of truthful, accurate answers, this dataset challenges models on their ability to discern and reproduce factually correct information [Lin et al., 2021].
- **OpenBookQA (OBQA)**: OpenBookQA requires understanding a wide array of facts and concepts, thereby evaluating the model's broader knowledge and reasoning skills [Mihaylov et al., 2018a].
- **LogiQA-2**: This dataset involves logical reasoning, testing the model's capability to understand and apply logical constructs and principles [Liu et al., 2023a].

Each dataset in these subsets is carefully selected to challenge and evaluate specific aspects of natural language processing models, ranging from domain-specific knowledge in science, medicine, and mathematics, to broader skills like common sense reasoning and general language understanding.

Filtering Details for Synthetic Data

When generating synthetic paraphrases using language models, we occasionally encounter the challenge of extraneous introductions in the generated outputs. Such paraphrases might commence with phrases like "Here's a paraphrase...", "The following..." or even contain keywords such as "high-quality English". To mitigate this, we've developed a method to filter and refine the synthetic outputs.

Methodology The primary function, `remove_unwanted_p` starts by splitting the input data into individual sentences. If the first sentence contains delimiters such as "\n\n" (indicating a new paragraph) or ":", the function checks the segment preceding the delimiter for the aforementioned unwanted elements. If these elements are detected, the preceding segment is removed. The entire revised content is then reconstructed and returned. In cases where

no modifications are applicable, but we still have the flagged keywords, we remove the paraphrase completely. To achieve this:

1. Split the input data into individual sentences using the NLTK's sentence splitter function.
2. Examine the first sentence for the presence of delimiters.
3. If a delimiter is detected, check the preceding segment for unwanted elements.
4. If unwanted elements are found, discard the preceding segment (before an occurrence of `"\n\n"` or `":"`).
5. Modify and return the filtered paragraph.

Based on manual inspection we found that the error rate (occurrence of sentences with unwanted elements) after the modification is less than 0.1%.

Properties of Synthetic Corpus

To understand the properties of synthetic data generated from the rephrase model that lead to better pre-training performance, we compare the semantic similarity, syntactic complexity, and diversity between synthetic data, C4 data, and data from the Pile. Our primary focus is answering the following questions about synthetic data: (i) Do models trained on synthetic data perform better due to information leakage from the rephrase model? (ii) Does the rephrase model accurately capture multiple styles? (iii) What attributes of synthetic data make it high quality? Our investigation helps address what data is beneficial for better generalization to specific domains, and quantify the importance of data variability and quality.

Experimental Setup We take a subset of the first 1000 documents from each of the datasets. For synthetic comparisons with real C4 data, we take pairs of samples, while for Pile subsets, we take the first 1000 samples from the test subset. When computing dataset quality statistics, we remove outliers more than two standard deviations in metric value. When the number of samples from the Pile subset was fewer than 1000, we split samples. Figures with distributions use a Gaussian Kernel Density Estimator (KDE) to construct distributions for statistics from 1000 values.

Semantic Properties In Section 13.5, we compared pairs of examples of synthetic and real data to confirm the performance gain is not attributed to knowledge leakage from the rephrase models using a pre-trained BERT model trained with SimCSE objective [Gao et al., 2021] for medium and qa prompts in Figure 13.5(a) and (b). We additionally compare the similarity of synthetic rephrases and actual rephrases using the MRPC corpus in Figure H.1(c). We denote this additional comparison by RealP (real paraphrase), while maintaining comparison of splits of

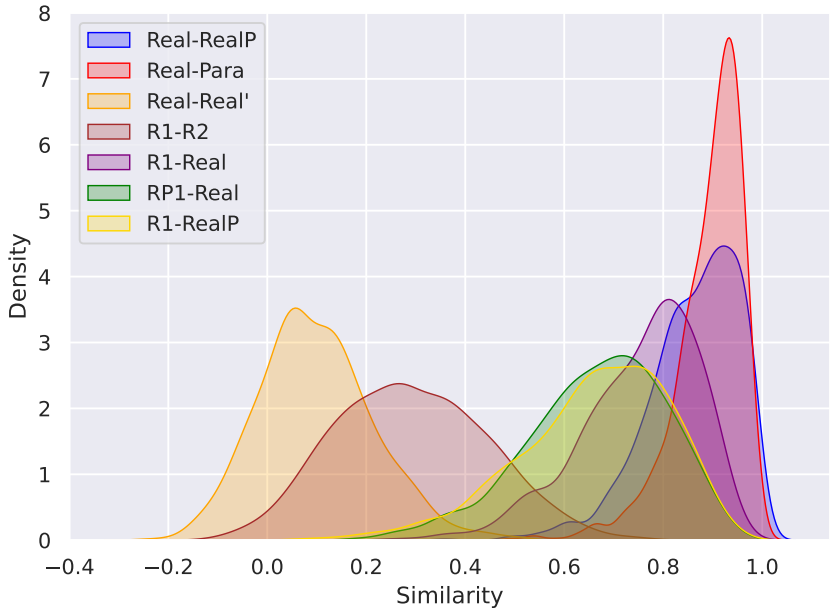
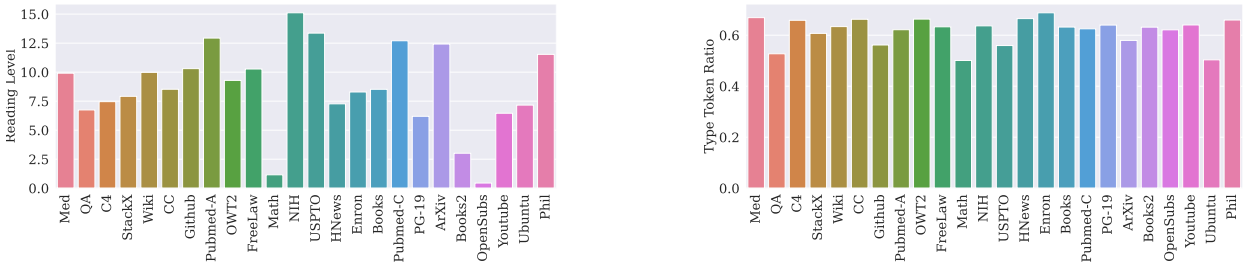


Figure H.1: Cosine similarity medium synthetic MRPC rephrases

the sentence: R1 and R2. Synthetic rephrases have similar cosine similarity on average and lower spread compared with true rephrases according in the MRPC corpus.

As the semantic information is similar between C4 and our synthetic data, we further investigate stylistic differences in the data. Figure H.2(a) shows the Flesch–Kincaid reading levels for different rephrase styles, and the Pile. Our findings indicate that C4 is on the low end of reading level (7-8). In contrast, medium increases the reading level to 10, and qa synthetic variants further reduces the reading level to 6. Medium synthetic data matches the reading level of Wikipedia, and other high reading level datasets yielding better performance on these domains. On QA synthetic data, we observe reduced reading level. This is because we observed that sentences are typically split into question and answer leading to shorter setnences compared with in the original text and medium style rephrases. This leads to lower metric values for many of the metrics. For type token ratio, we note that the diversity is quite similar between medium and most subsets of the Pile. The QA dataset has particularly low TTR matching ubuntu, github, and math as these are more similar to QA format datasets and have heavy repetition of the Question, and Answer format.

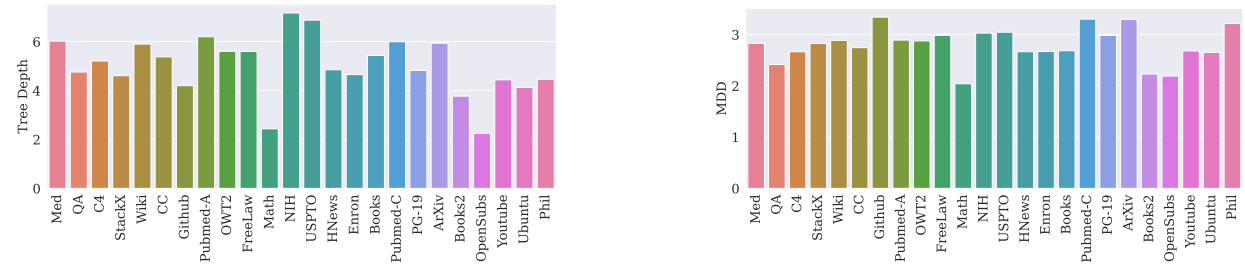
Syntactic Properties Finally, we compare the mean tree depth (measured by the mean over setences of the depth of the dependency tree), and mean dependency distance (measured as the average dependency distance of any pair of words within a sentence) in Figure H.3, which have been shown to be good measures of syntactic difficulty [Futrell



(a) Flesch-Kincaid Reading Level

(b) Type token ratio

Figure H.2: Comparison of readability and diversity (ttr) of synthetic data compared with C4 and different subsets of the Pile.



(a) Tree Depth

(b) Mean Dependency Distance

Figure H.3: Comparison between synthetic and real data from the C4 corpus showing that synthetic data have higher syntactic complexity indicated by higher average tree depth, and higher mean dependency distance (MDD).

et al., 2015, Gibson et al., 2000, Oya, 2021]. We find similar trends as for reading level and TTR diversity where mediums tyle increase depth, mdd, and syntactic complexity in general. We find again that QA style reduces this complexity.

Evaluation Metrics

The metric utilized for evaluation is the *macro token level perplexity*. Given a batch of encoded texts, the perplexity at the token level was computed as follows:

Given the accumulated loss over the entire dataset, denoted as L, and the total number of tokens, represented by T, the macro token-level perplexity, denoted as $\mathcal{P}$, is calculated as:

$$\mathcal{P} = \exp\left(\min\left(20, \frac{L}{T}\right)\right) \tag{H.1}$$

Where:

- exp is the exponential function.
- L is the cumulative loss over all shifted logits and labels in the dataset.
- T is the total number of tokens in the dataset.

The value of 20 acts as an upper limit to stabilize the metric in cases of high loss values.

Limitations

Cost Analysis *Should you generate synthetic data, or just train longer on real data?*

The applications of T-MARS lies in both paradigms—(i) low-resourced data settings such as a language model for Finnish language [Luukkonen et al., 2023], and (ii) data-rich settings such as training on the common crawl. In the former, there is no alternative option of naively gathering more data, and hence, synthetic data is a natural solution that should outperform training on in-domain data alone. However, there is a significant interest in training language models on English, or more broadly, general web data. Is using synthetic data a viable option even in this paradigm?

Before, we dive into the feasibility of pre-training on synthetic data, we should acknowledge the results of Table **??**. The TinyLlama model trained for 3 Trillion tokens also underperforms a model jointly trained on real and synthetic data. In fact, it performs quite comparably to the models that were trained for 300B tokens on just real data as well. This suggests that the ceiling for improvement by training for longer may not be that high (for a model of size 350M/1.3B parameters; larger models may benefit from training for longer).

To analyze this cost trade-off, we compare the cost of generating synthetic data, versus that of training a language model on extra data. For our synthetic data generation experiments, we use the vLLM [Kwon et al., 2023] library for fast generation. In particular, we are able to generate 3M tokens per hour on a single A100 when using the Mistral-7B. Generating 85B tokens (as in our

work) accounts for about 25K GPU hours.

In comparison, on 64 A100s, we achieve a throughput of 0.5M tokens per second. Assuming training for 300B tokens, would mean 256 GPU days, accounting for about 6k GPU hours to train a single model. On the contrary, training a 13B model would take about 30K GPU hours. At the scale of training a 13B model, reducing the training cost by 3-10x can incorporate the cost overhead of training with synthetic data in a single run.

While the cost of generating high quality data is still relatively high, two important sources of improvement impact this cost analysis. First, if we use the Qwen-1.8B model [Bai et al., 2023b] for rephrasing, we are able to get a 3x higher token throughput. As seen in our preliminary results in Fig 13.7.1, the model pre-trained on rephrases generated by Qwen model performs comparably to that by the Mistral model. This reduces the cost of generation by 3x. More recent work in speculative decoding [Liu et al., 2023b] and optimized inference [Xia et al., 2024] suggest that we can leverage another 3-5x improvement in the generation cost. Hence, indeed, even at the scale of just 1.3B parameter model training, we can already improve upon the cost of pre-training using just real data.

Two additional important advantages of synthetic data generation that could not be accounted for in the discussion above:

1. The cost of synthetic data generation is a one-time investment, and we may train many models of varying scales once the data is generated.
2. Data generation is 100% parallelizable, whereas training requires the availability of a big cluster with fast inter-node connectivity. This is much more expensive. On the other hand, generation can be thought of as a side process that can fill in the empty GPUs in any large-scale compute cluster, and runs on single GPU machines.

Diversity of Synthetic Generations Another limitation is enforcing the diversity in the generated data. This diversity comes from both the "style" and the "knowledge" contained in the generated data. Recent works [Li et al., 2023c,d] used a selection of topics, or scenarios to seed the model to generate novel texts. Still, a recent study by Padmakumar et al. [2023] showed that using language models for AI-assisted writing tends to reduce content diversity, particularly when using instruction-tuned models. While we used the paradigm of rephrasing specifically to mitigate the issues pertaining to the diversity of novel content generation, it remains for future work to assess the presence (or lack of) and impact of content diversity in paraphrase models.

Additional Results for Smaller Model and Token Sizes

Results for 350M Models Trained for 75B Tokens We train models at smaller scales and demonstrate improvement. In particular we train a 350M GPT-2-medium architecture for a total of 75B tokens. We show Pile

Dataset (Real Tok.)	ARC-E	BoolQ	Wino.	PIQA	HellaSwag	TruthfulQA	OBQA	LogiQA	[redacted]
C4-18B	43.7	52.8	53.0	70.1	43.9	37.8	29.6	26.5	
C4-75B	43.9	53.4	51.6	71.2	44.2	39.0	29.6	25.7	
QA+C4-18B	47.3	60.7	52.2	71.2	44.6	40.0	31.0	26.3	
Med+C4-18B	43.6	57.3	53.6	70.8	43.4	36.9	31.0	26.2	

Dataset (Real Tok.)	ARC-C	SciQ	PubMedQA	MathQA	MMLU	[redacted]
C4-18B	25.3	66.6	50.6	22.0	23.1	
C4-75B	24.3	66.8	46.4	22.1	23.0	
QA+C4-18B	27.3	69.8	56.0	21.4	22.9	
Med+C4-18B	6.1	67.6	53.6	21.7	23.1	

Table H.2: Medium Sized Model Evaluation: (Left) Evaluation of 350M parameter LLMs trained for 75B tokens on General understanding Tasks. This table shows the performance across various datasets, focusing on general reasoning, language understanding, and common sense comparing training. (Right) Evaluation Specialized Knowledge Tasks. This table presents the performance on tasks that require specific domain knowledge such as science, medicine, mathematics, and logic.

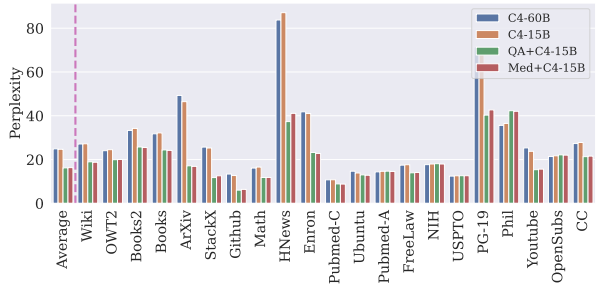


Figure H.4: Perplexity across all domains of the Pile comparing combining multiple styles of synthetic data. Models are 350M parameters trained for a total of 75B tokens.

perplexity averaged across the 21 domains is much lower than for that of the model trained only on C4 in Figure H.4, and even lower than 1.3B models trained only on C4 in Figure 13.7.1. We also show an increase of 2% across general understanding language tasks, and roughly 2 – 3% on specialized knowledge tasks in Table H.2 when adding QA rephrases. We also experimented with medium rephrases at this smaller scale. Our findings indicate that the high quality provided by medium rephrases improves performance over only C4, however matching the style as indicated by QA rephrase performance further improves performance.

Results for 1.3B Models Trained for 150B Tokens We additionally train 1.3B GPT-2-XL models at 150B tokens, reducing the number of steps by half. We show Pile perplexity averaged across the 20 domains is much lower than for that of the model trained only on C4 in Figure H.5, and even lower than 1.3B models trained only on C4 in Figure 13.7.1 for twice as long. We also show an increase of 2% across specialized knowledge tasks, and roughly 2% on general understanding tasks in Table H.3 when adding QA rephrases. We also experimented with medium rephrases at this smaller scale, and report similar findings consistent with other small-scale experiments.

Ablations

We further ask the following Research Questions (RQs) to investigate in a finer granularity for how to enhance performance with synthetic data.

Data Combination Analysis

Dataset (Real Tok.)	ARC-E	BoolQ	Wino.	PIQA	HellaSwag	TruthfulQA	OBQA	LogiQA
C4-35B	51.6	55.2	56.1	75.2	58.2	36.0	33.6	27.8
C4-150B	52.2	54.4	56.4	75.6	58.6	34.3	35.8	27.1
Med+C4-35B	53.2	57.0	55.7	74.8	57.6	36.5	33.2	26.2
QA+C4-35B	55.1	63.3	55.7	75.6	57.9	41.4	34.0	26.1

Dataset (Real Tok.)	ARC-C	SciQ	PubMedQA	MathQA	MMLU
C4-35B	28.0	74.3	55.0	22.0	24.3
C4-150B	28.5	75.8	55.4	22.6	25.4
Med+C4-35B	29.6	74.3	46.2	22.9	25.2
QA+C4-35B	30.3	76.8	62.2	23.0	26.1

Table H.3: Left) Evaluation of ∼ 1.3B parameter LLMs trained for 150B tokens on General Understanding Tasks. This table shows the performance across various datasets, focusing on general reasoning, language understanding, and common sense comparing training . **Right** Evaluation on Specialized Knowledge Tasks. This table presents the performance on tasks that require specific domain knowledge such as science, medicine, mathematics, and logic.

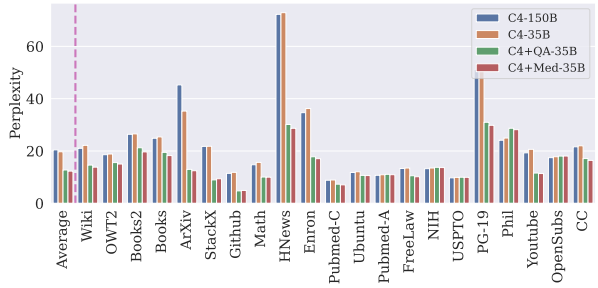


Figure H.5: Perplexity across all domains of the Pile comparing combining multiple styles of synthetic data. Models are 350M parameters trained for a total of 75B tokens.

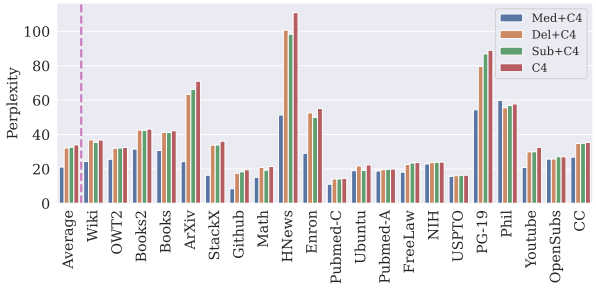


Figure H.6: Is re-phrasing same as any augmentation? We compare perplexity on the Pile for different augmemntation strategies. 350M parameter models are trained for a total of 15B tokens. T-MARS (Medium + C4) performs significantly better than traditional augmentations.

RQ4: Does a combination of multiple synthetic datasets improve performance? We measure the impact of combining multiple synthetic styles with C4 for training. We consider two variants: combining in a 1:1 ratio meaning that there are two copies of C4 to match two synthetic styles (medium and QA), and 1:2 ratio which combines only one instance of the C4 dataset. For zero-shot QA tasks, our finding in Table **??** indicate lower performance than combining only QA and C4 data. Evaluations over the Pile are shown in Figure 13.7.1. We notice that both the 'Q/A' and 'Wikipedia' paraphrases help improve performance on certain domains. For example, 'Stackexchange', that has lots of question-answers benefits from the presence of synthetic data in Q/A style. Overall, we note that there is a small improvement on the average perplexity on the Pile by combining multiple styles.

Method Ablations

RQ5: Does synthetic data improve over augmentations? Are the gains observed by pre-training on synthetic data the same as pre-training with augmentations? In order

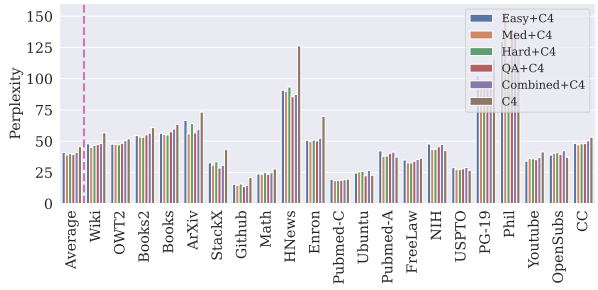


Figure H.7: Impact of style of synthetic rephrases: Perplexity across all domains of the Pile comparing different styles of synthetic data. We train 128M parameter models for 3B tokens.

to test this, we consider two popular text augmentation baselines—synonym replacement and random deletion using the NL-Augmenter library [Dhole et al., 2023]. We pre-train a 350M parameter model for 15B tokens in order to conduct this set of experiments. The total pool size is only about 1.5B tokens, meaning that the model would have to repeat data around 10 times during the pre-training phase, unless augmented over. As seen in the perplexity analysis in Figure H.6, the models trained on augmented data perform significantly worse than those trained on combinations of real and synthetic data. This suggests that synthetic data enhances the learning process, and is not merely another form of augmentation.

RQ6: How does the style of synthetic data impact performance on specialized domains? We compare the performance of various models trained on different styles of synthetic data. In particular, we generate four styles of synthetic data (easy, medium, hard, and Q/A) and evaluate the performance of training on combinations of each style across Pile subsets. The prompts to generate these synthetic data styles are outlined in Appendix H. Results corresponding to generations from a Vicuna-v1.3 model, and for a 128M model trained for 3B tokens are summarized in Figure H.7. We see that training with combinations of real C4 and synthetic data matching the style of the domain at evaluation improves performance. However, we find that no single synthetic data style performs the best across all domains, resulting in similar performance across training with combinations of real C4 data and each synthetic style variant. While knowing the best synthetic style to pre-train an LLM is impractical, an oracle that selects the best synthetic style across all domains will improve perplexity by 16%—indicating the importance of training with diverse data styles for LLM generalization, even when the underlying knowledge stays the same.

LLM Leaderboard Few-shot Results

In our main experiments in Section 13.3 we demonstrate that LLMs trained with synthetic rephrases are a better backbone for zero-shot question-answering tasks as the model learns the question-answer format and style during pre-training. In this section, we show that improvements from pre-training on synthetic rephrases are still present even in few-shot settings where the model has access to test samples. To study few-shot performance, we evaluate on six tasks

present in the OpenLLMLeaderboard[1]:

1. ARC-Challenge (25 shot)
2. HellaSwag (10 shot)
3. MMLU (5 shot)
4. Truthful-QA (5 shot)
5. Winogrande (5 shot)
6. GSM8k (5 shot)

We evaluate two models trained for 300B and 350B tokens corresponding to roughly 85B and 100B unique C4 tokens respectively. Our findings show substantial improvements on the ARC-challenge benchmark, and Truthful-QA conssitent in the zero-shot settings and comparable performance across other datasets. Our models also perform better than the publicly released Falcon-1.3B model trained on the Refined Web dataset, and the Pythia-1.4B model, which was trained on Pile.

[1] https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

Dataset	ARC	Hellaswag	MMLU	TruthfulQA	WinoGrande	GSM8K	[redacted]
C4	31.7	62.1	26.7	33.4	57.9	0.9	
Falcon-RW	35.1	63.6	25.3	36.0	62.0	0.5	
Pythia-1.4b-Pile	32.7	55.0	25.6	38.7	57.3	0.8	
TinLlama	33.9	60.3	26.0	37.3	59.5	1.4	
QA+C4-85B (300K)	36.4	60.9	25.5	40.6	59.4	0.4	
QA+C4-100B (350K)	35.5	60.5	26.8	40.6	61.3	0.3	

Table H.4: 1.3B 300K LLM Leaderboard Eval. Evaluation for T-MARS is done on a single seed (1234).

Rephrase Prompt Templates

We detail the prompts given to the Mistral-7B model to generate synthetic versions of the C4 dataset in specific styles. *Note: there are slight variations in the prompt that were used for other frozen LLMs, and no prompt was used for the T5 model.*

Easy Style A style designed to generate content understandable by toddlers.

```
A chat between a curious user
and an artificial intelligence
assistant. The assistant gives
helpful, detailed, and polite
answers to the questions. USER:
For the following paragraph
give me a paraphrase of
the same using a very small
vocabulary and extremely simple
sentences that a toddler will
understand:
```

Hard Style A style designed to generate content comprehensible primarily to scholars using arcane language.

```
A chat between a curious user
and an artificial intelligence
assistant. The assistant gives
helpful, detailed, and polite
answers to the questions. USER:
For the following paragraph
give me a paraphrase of the
same using very terse and
abstruse language that only an
erudite scholar will understand.
Replace simple words and
phrases with rare and complex
ones:
```

Medium Style A style designed to generate content comparable to standard encyclopedic entries.

```
A chat between a curious user
and an artificial intelligence
assistant. The assistant
gives helpful, detailed, and
polite answers to the questions.
USER: For the following
paragraph give me a diverse
paraphrase of the same in high
quality English language as in
sentences on Wikipedia:
```

Q/A Style A style intended to convert narratives into a conversational format.

```
A chat between a curious user
and an artificial intelligence
assistant.  The assistant
gives helpful, detailed, and
polite answers to the questions.
USER: Convert the following
paragraph into a conversational
format with multiple tags
of "Question:" followed by
"Answer:":
```

Rephrase Examples from the MRPC Corpus

Original

```
The stock rose $2.11, or about
11 percent, to close Friday at
$21.51 on the New York Stock
Exchange.

Revenue in the first quarter
of the year dropped 15 percent
from the same period a year
earlier.
```

Q/A Style

```
Question: What was the stock's
closing price on Friday?
Answer: $21.51 Question: How
much did the stock rise on
Friday? Answer: $2.11 or
about 11 percent.

Question: What was the revenue
drop in the first quarter
compared to the same period
last year? Answer: The
revenue dropped 15 percent.
```

Medium Style

```
The stock experienced an
increase of approximately 11
percent, closing at $21.51 on
the New York Stock Exchange on
Friday, with a rise of $2.11.

During the initial three months
of the current year, there
was a 15 percent decrease
in revenue compared to the
corresponding quarter of the
previous year.
```

Rephrase Examples from C4 Corpus

Original

First round on stress at work survey. Answering the questionnaire is voluntary and all answers will be saved anonymously. Please fill in this questionnaire only if you have some work experience part-or full time. Otherwise you will not be able to answe some of the questions! Here is a the link to all language version.

Not that there's a thing wrong with frozen burgers. The key here is the meat seasonings, which are pretty strong and spicy and just GOOD, something else I think is really necessary in a turkey burger because ground turkey otherwise can be kind of flavorless. You'll need ground turkey, onion powder, chili powder, salt, pepper, and cayenne pepper for the burgers. Then the mayo takes garlic and onion. Then we need buns, clearly, swiss cheese, lettuce, and onion. I LOVE tomatoes but sometimes find that they get in the way of other flavors, so I left them off of this burger. Add them if you'd like to your array of toppings! First, we'll make the mayo. Grate the garlic directly into the mayo, add a pinch of salt, and squeeze in the lemon juice. Stir. Done! I love this. Then, we'll work on the burgers. Preheat a large skillet to medium-high heat with some olive oil, preheat the broiler to high, then add all the spices to the ground turkey.

Whether you like your velvet crushed, vibrant or head-to-toe, there's really no denying the sheer luxe and elegance of this timeless textile. Not only is it super stylish, it can actually be so wearable for day-to-day wear. Yes, really! This year it's all about embracing fun gem-toned velvety pieces. Long gone are the days when velvet was solely associated with dark moody shades of navy and black. Below we've rounded up the most covetable velvet pieces on the high street right now. We're already coming up with outfit ideas! Are you completely obsessed or beyond bored of it?

Save up to $8,086 on one of 1,258 Chrysler 200s near you. Find your perfect car with Edmunds expert and consumer car reviews, dealer reviews, car comparisons and pricing tools. We have 4,850,420. Research 2015 Chrysler 200 Sedan 4D 200C I4 prices, used values & 200 Sedan 4D 200C I4 pricing, specs and more. Many years ago, we wrote about the stalling problem with the 2011 Chrysler 200, and believe it or not, we still receive an occasional call regarding the problem.However, a much larger issue has monopolized the phone lines as of late 2015 Chrysler 200 transmission problems leaving drivers with check engine lights, harsh shifting, and the occasional loss of power. The 2015 Chrysler 200 can fetch a premium for its style and its horsepower-but rear-seat room and handling are better bargains elsewhere. Find out why the 2015 Chrysler 200 is rated 8.4 by The. Don't know where to find the perfect rims for your 2015 Chrysler 200 CARiD.com stores a massive selection of 2015 Chrysler 200 wheels offered in myriads of design and finish options, including chrome, black, silver, and so much more.

Medium Style	In the initial stage of t survey on work-related st participation is optional all responses will be kep confidential. It is impo to note that only those w work experience, whether part-time or full-time, w be able to accurately res to certain questions. To complete the questionnair please use the following which is available in mul languages.	Frozen burgers are not inherently flawed, but the meat seasonings are a crucial element that elevates the flavor of the dish. Ground turkey can be quite bland without the right blend of spices, so it is essential to add onion powder, chili powder, salt, pepper, and cayenne pepper to the mixture. Additionally, garlic and onion should be used in the mayonnaise, which is necessary for the burger's topping. To make the mayo, grate the garlic directly into the mixture, add a pinch of salt, and squeeze in lemon juice. Mix well. The mayonnaise is a delicious addition to the burger topping. Now, let's work on the burgers. Heat a large skillet over medium-high heat with some olive oil, and preheat the broiler to high. Add the seasonings to the ground turkey and mix well. The burgers should be cooked on the skillet for about 5 minutes on each side, then transferred to the broiler to cook for an additional 2-3 minutes. The burgers are ready to be assembled on buns, along with swiss cheese, lettuce, and onion. Tomatoes can be added as a topping, but they may overpower other flavors.

The luxurious and elegant nature of velvet cannot be denied, whether it is crushed, vibrant, or head-to-toe. This timeless textile is not only fashionable but also wearable for daily wear. In recent years, there has been a shift towards embracing fun gem-toned velvety pieces, breaking away from the traditional association of velvet with dark moody shades of navy and black. Below are some of the most desirable velvet pieces available on the high street today, and we are already envisioning outfits around them. Are you completely enamored with velvet or simply beyond bored with it?

Discover savings up to $8,086 on one of 1,258 Chrysler 200s near you. Get expert and consumer car reviews, dealer reviews, car comparisons, and pricing tools from Edmunds. We have 4,850,420 listings for 2015 Chrysler 200 Sedan 4D 200C I4 prices, used values, and more. While we wrote about the stalling problem with the 2011 Chrysler 200, we still receive occasional calls regarding the issue. However, a more significant issue has taken over the phone lines recently - 2015 Chrysler 200 transmission problems that leave drivers with check engine lights, harsh shifting, and occasional loss of power. Despite the premium price for its style and horsepower, rear-seat room and handling are better bargains elsewhere. The 2015 Chrysler 200 is rated 8.4 by The Car Connection. If you’re looking for the perfect rims for your 2015 Chrysler 200 CARiD.com offers a vast selection of 2015 Chrysler 200 wheels in various design and finish options, including chrome, black, silver, and more.

Q/A Style

Question: Is participating in the stress at work survey mandatory? Answer: No, it i voluntary. Question: Will a answers be saved anonymously? Answer: Yes, they will. Question: Who is eligible to take the questionnaire? Answ Only individuals with work experience, whether part-time or full-time. Question: What happens if someone is not eligible to take the questionnaire? Answer: They will not be able to answer so of the questions. Question: Can you provide a link to all language versions of the questionnaire? Answer: Yes, here is the link.

Question: What makes these frozen burgers so good? Answer: The meat seasonings are strong and spicy, and they give the burgers a good flavor. Question: Do you think ground turkey can be flavorless? Answer: Yes, ground turkey can be flavorless, but adding spices and seasonings can help. Question: What do you need to make these turkey burgers? Answer: Ground turkey, onion powder, chili powder, salt, pepper, and cayenne pepper for the burgers. Then the mayo takes garlic and onion. Buns, swiss cheese, lettuce, and onion are also needed. Question: What about toppings? Answer: Add tomatoes if you'd like to your array of toppings. Question: How do you make the mayo? Answer: Grate the garlic directly into the mayo, add a pinch of salt, and squeeze in the lemon juice. Stir. Done! Question: How do you work on the burgers? Answer: Preheat a large skillet to medium-high heat with some olive oil, preheat the broiler to high, then add all the spices to the ground turkey.

Question: What do you think of velvet? Answer: Whether you like your velvet crushed, vibrant or head-to-toe, there's really no denying the sheer luxe and elegance of this timeless textile. Question: Is velvet super stylish? Answer: Yes, really! This year it's all about embracing fun gem-toned velvety pieces. Question: Was velvet solely associated with dark moody shades of navy and black? Answer: Long gone are the days when velvet was solely associated with dark moody shades of navy and black. Question: Do you have any covetable velvet pieces on the high street right now? Answer: Below we've rounded up the most covetable velvet pieces on the high street right now. Question: Are you completely obsessed or beyond bored of it? Answer: We're already coming up with outfit ideas!

Question: How can I save money on a Chrysler 200? Answer: Save up to $8,086 on one of 1,258 Chrysler 200s near you. Find your perfect car with Edmunds expert and consumer car reviews, dealer reviews, car comparisons and pricing tools. Question: What are the problems with the 2015 Chrysler 200? Answer: We have 4,850,420. Research 2015 Chrysler 200 Sedan 4D 200C I4 prices, used values & 200 Sedan 4D 200C I4 pricing, specs and more. Many years ago, we wrote about the stalling problem with the 2011 Chrysler 200, and believe it or not, we still receive an occasional call regarding the problem. However, a much larger issue has monopolized the phone lines as of late 2015 Chrysler 200 transmission problems leaving drivers with check engine lights, harsh shifting, and the occasional loss of power. Question: What are the benefits of buying a 2015 Chrysler 200? Answer: The 2015 Chrysler 200 can fetch a premium for its style and its horsepower-but rear-seat room and handling are better bargains elsewhere. Question: How is the 2015 Chrysler 200 rated? Answer: It's rated 8.4 by The. Question: Where

Chapter I

BeyondWeb

Appendix

Additional Training Details For all trainings, we use AdamW with β_1=0.9, β_2=0.95, a learning rate of $5e^{-4}$, and a weight decay of $1e^{-7}$. A linear warmup schedule was applied with 4K steps for the 1B model and 16K steps for the 3B and 8B models. Training was performed using fully sharded data parallelism (FSDP), with a batch size of 512, and a context length of 2048.

Evaluation Tasks We evaluate using lighteval [Habib et al., 2023] on a suite of 14 evaluation datasets that includes the 8 used by FineWeb [Penedo et al., 2024] and additional datasets that are well-established and demonstrate above-chance performance and monotonic improvement during training.

- **ARC-Challenge** [Clark et al., 2018b]: A dataset of 2,590 genuine grade-school level, multiple-choice science questions, designed to promote research in advanced question-answering. It challenges AI capabilities by requiring reasoning and comprehension beyond standard retrieval methods.
- **ARC-Easy** [Clark et al., 2018b]: Similar to ARC-Challenge, but consists of 7,787 easier questions.
- **BoolQ** [Clark et al., 2019]: A question answering dataset for yes/no questions containing 15,942 examples. These questions are naturally occurring and generated in unprompted and unconstrained settings.
- **CommonsenseQA** [Talmor et al., 2019]: A multiple-choice question answering dataset containing 12,102 questions requiring different types of commonsense knowledge.
- **COPA** [Roemmele et al., 2011]: Designed to assess commonsense causal reasoning through 1000 questions where each question consists of a premise and two alternatives.
- **HellaSwag** [Zellers et al., 2019]: A benchmark dataset for commonsense natural language inference that challenges state-of-the-art models with context and endings that are easy for humans but difficult for machines.
- **MMLU** [Hendrycks et al., 2021c]: A benchmark evaluating capabilities across 57 subjects, including mathematics, history, and law.
- **OpenbookQA** [Mihaylov et al., 2018b]: Contains 5,957 multiple-choice questions designed to probe understanding

of core science facts and their applications.

- **PIQA** [Bisk et al., 2020b]: Evaluates physical commonsense reasoning through everyday scenarios, with about 20,000 question-answer pairs.
- **RACE-High** [Lai et al., 2017]: Contains 69,395 questions from 19,527 passages from English examinations for Chinese high school students.
- **RACE-Middle** [Lai et al., 2017]: Consists of 28,293 questions from 8,718 passages targeted at middle school students.
- **SciQ** [Welbl et al., 2017]: Contains 13,679 crowdsourced science exam questions covering topics such as physics, chemistry, and biology.
- **SIQA** [Sap et al., 2019]: Contains over 38,000 multiple-choice questions about everyday social interactions.
- **WinoGrande** [Sakaguchi et al., 2020]: Contains 44,000 problems designed to evaluate commonsense reasoning in AI systems.

Quantifying Conversational Content in Web-Scale Corpora To understand the proportion of conversational text in the web data, we conducted an experiment using a classification task on a sample of data from the C4 dataset. The goal was to assess what proportion of naturally occurring web data aligns with chat-based use cases.

For this, we provided the GPT-4 model with a classification task to label text as either conversational or non-conversational based on the following criteria for classification.

Classification Task: Conversational vs. Non-Conversational

Your task is to classify text as either conversational (1) or non-conversational (0).
Conversational text (1) shows:

- Back-and-forth exchanges between participants
- Questions followed by relevant answers

Non-conversational text (0) typically:

- Presents information in a one-way format
- Lacks interaction between participants
- Contains formal or encyclopedic writing
- Focuses on describing or explaining without dialogue

Rate each text as exactly 0 or 1. Return only the number.
Examples:

- **Text:** "The mitochondria is the powerhouse of the cell. It produces energy through cellular

respiration. This process involves multiple steps including glycolysis and the Krebs cycle."
Output: 0

- **Text:** "A: Hey, I'm having trouble with my code. The function keeps returning null. B: Can you share the error message you're getting? A: Here's the stack trace: [error details] B: Ah, I see the issue. You need to initialize the variable first. A: That fixed it, thanks!"
Output: 1

- **Text:** "Q: What's the capital of France? A: The capital of France is Paris. Q: What's its population? A: Paris has a population of about 2.2 million people."
Output: 1

- **Text:** "To install Python, first download the installer from python.org. Run the executable and follow the installation wizard. Make sure to check the 'Add Python to PATH' option during installation."
Output: 0

- **Text:** "JavaScript was created by Brendan Eich in 1995 while he was working at Netscape Communications Corporation. The language was originally designed for client-side web development."
Output: 0

- **Text:** "User1: Did anyone solve the memory leak issue? User2: Yes, it was related to the event listeners User1: How did you fix it? User2: We added a cleanup function in the useEffect hook User1: Got it, I'll try that"
Output: 1

- **Text:** "The Renaissance was a period in European history marking the transition from the Middle Ages to modernity and covering the 15th and 16th centuries."
Output: 0

- **Text:** "- Can we push back the deadline? - I need to check with the team first. When would you need it by? - Would next Friday work? - Yes, that should be fine. I'll update the project timeline."
Output: 1

Classify the following text:

The experiment was performed on a random sample of 10,000 text examples from the C4 dataset, with a final result indicating that approximately 2.7% of the sampled text was classified as conversational. This proportion is reflective of the prevalence of conversational data within a large-scale corpus drawn from the web.

Additional Results

Figure I.1 shows that BeyondWebconsistently leads throughout training, achieving the highest final accuracy at all scales. As shown in Figure I.2, the 3B BeyondWebmodel even surpasses most 8B baselines under the same token budget. Table **??** and Table **??** confirm broad 0-shot and 5-shot improvements across 14 tasks, with BeyondWeboutperforming other methods on most of benchmarks at each scale.

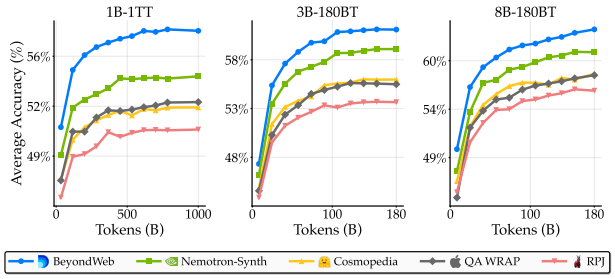


Figure I.1: Learning curves across model scales demonstrate our consistent performance advantages. The plots show training dynamics for 1B-1TT, 3B-180BT, and 8B-180BT configurations. Average Accuracy (%) is computed as the mean score across 14 standard benchmarks, averaged over both 0-shot and 5-shot settings. BeyondWeb(blue) maintains superior performance throughout training across all scales, achieving 57.4%, 60.8%, and 63.7% accuracy respectively.

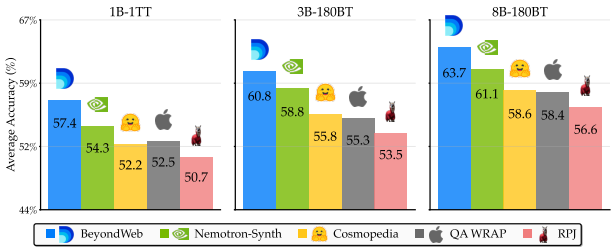


Figure I.2: Performance comparison showing BeyondWeb's significant advantages over state-of-the-art baselines across three model scales. Average Accuracy (%) is computed as the mean score across 14 standard benchmarks, averaged over both 0-shot and 5-shot settings. We highlight that the 1B BeyondWebmodel outperforms all but one 3B baseline, and the 3B BeyondWebmodel outperforms all but one 8B baseline, despite the 3B and 8B models being trained for the same token budget (180B tokens).

Bibliography

Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing, 2001. URL https://aclanthology.org/W01-0500.

M. Abdin, J. Aneja, H. Awadalla, A. Awadallah, A. A. Awan, N. Bach, A. Bahree, A. Bakhtiari, J. Bao, H. Behl, et al. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*, 2024a.

M. Abdin, J. Aneja, H. Behl, S. Bubeck, R. Eldan, S. Gunasekar, M. Harrison, R. J. Hewett, M. Javaheripi, P. Kauffmann, J. R. Lee, Y. T. Lee, Y. Li, W. Liu, C. C. T. Mendes, A. Nguyen, E. Price, G. de Rosa, O. Saarikivi, A. Salim, S. Shah, X. Wang, R. Ward, Y. Wu, D. Yu, C. Zhang, and Y. Zhang. Phi-4 technical report. 2024b. URL https://arxiv.org/abs/2412.08905.

M. Afifi. 11k hands: gender recognition and biometric identification using a large dataset of hand images. *Multimedia Tools and Applications*, 2019. doi: 10.1007/s11042-019-7424-8. URL https://doi.org/10.1007/s11042-019-7424-8.

AI@Meta. Llama 3 model card, 2024. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

S. Alemohammad, J. Casco-Rodriguez, L. Luzi, A. I. Humayun, H. Babaei, D. LeJeune, A. Siahkoohi, and R. G. Baraniuk. Self-consuming generative models go mad. *arXiv preprint arXiv:2307.01850*, 2023.

R. Aljundi, P. Chakravarty, and T. Tuytelaars. Expert gate: Lifelong learning with a network of experts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3366–3375, 2017.

L. B. Allal, elie, A. Marafioti, A. Lozhkov, Q. Lhoest, M. Noyan, M. Farré, G. M. Blázquez, N. Tazi, yousan, A. Bartolome, E. Ferdman, J. Isbarov, vb, and L. Cheng. *huggingface/smollm*. 3 2025. URL https://github.com/huggingface/smollm.

Z. Allen-Zhu and Y. Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023.

A. Amini, S. Gabriel, S. Lin, R. Koncel-Kedziorski, Y. Choi, and H. Hajishirzi. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2357–2367, 2019.

M. Andriushchenko. Adversarial attacks on gpt-4 via simple random search. 2023.

R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen, E. Chu, J. H. Clark, L. E. Shafey, Y. Huang, K. Meier-Hellstern, G. Mishra, E. Moreira, M. Omernick, K. Robinson, S. Ruder, Y. Tay, K. Xiao, Y. Xu, Y. Zhang, G. H. Abrego, J. Ahn, J. Austin, P. Barham, J. Botha, J. Bradbury, S. Brahma, K. Brooks, M. Catasta, Y. Cheng, C. Cherry, C. A. Choquette-Choo, A. Chowdhery, C. Crepy, S. Dave, M. Dehghani, S. Dev, J. Devlin, M. Díaz, N. Du, E. Dyer, V. Feinberg, F. Feng, V. Fienber, M. Freitag, X. Garcia, S. Gehrmann, L. Gonzalez, G. Gur-Ari, S. Hand, H. Hashemi, L. Hou, J. Howland, A. Hu, J. Hui, J. Hurwitz, M. Isard, A. Ittycheriah, M. Jagielski, W. Jia, K. Kenealy, M. Krikun, S. Kudugunta, C. Lan, K. Lee, B. Lee, E. Li, M. Li, W. Li, Y. Li, J. Li, H. Lim, H. Lin, Z. Liu, F. Liu, M. Maggioni, A. Mahendru, J. Maynez, V. Misra, M. Moussalem, Z. Nado, J. Nham, E. Ni, A. Nystrom, A. Parrish, M. Pellat, M. Polacek, A. Polozov, R. Pope, S. Qiao, E. Reif, B. Richter, P. Riley, A. C. Ros, A. Roy, B. Saeta, R. Samuel, R. Shelby, A. Slone, D. Smilkov, D. R. So, D. Sohn, S. Tokumine, D. Valter, V. Vasudevan, K. Vodrahalli, X. Wang, P. Wang, Z. Wang, T. Wang, J. Wieting, Y. Wu, K. Xu, Y. Xu, L. Xue, P. Yin, J. Yu, Q. Zhang, S. Zheng, C. Zheng, W. Zhou, D. Zhou, S. Petrov, and Y. Wu. Palm 2 technical report, 2023. URL https://arxiv.org/abs/2305.10403.

Anthropic. Claude 3 model card, 2024. URL https://www.anthropic.com/news/claude-3-family.

A. Arditi and B. Chughtai. Unlearning via RMU is mostly shallow, July 2024. URL https://www.lesswrong.com/posts/6QYpXEscd8GuE7BgW/unlearning-via-rmu-is-mostly-shallow. AI Alignment Forum, informal research note.

D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017.

L. Atkins. Announcing the official launch of afm-4.5b. July 2025. URL https://www.arcee.ai/blog/announcing-the-official-launch-of-afm-4-5b.

W. Baek. Torchskeleton. https://github.com/wbaek/torchskeleton, 2019.

J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, B. Hui, L. Ji, M. Li, J. Lin, R. Lin, D. Liu, G. Liu, C. Lu, K. Lu, J. Ma, R. Men, X. Ren, X. Ren, C. Tan, S. Tan, J. Tu, P. Wang, S. Wang, W. Wang, S. Wu, B. Xu, J. Xu, A. Yang, H. Yang, J. Yang, S. Yang, Y. Yao, B. Yu, H. Yuan, Z. Yuan, J. Zhang, X. Zhang, Y. Zhang, Z. Zhang, C. Zhou, J. Zhou, X. Zhou, and T. Zhu. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023a.

J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023b.

S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, H. Zhong, Y. Zhu, M. Yang, Z. Li, J. Wan, P. Wang, W. Ding, Z. Fu, Y. Xu, J. Ye, X. Zhang, T. Xie, Z. Cheng, H. Zhang, Z. Yang, H. Xu, and J. Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, N. Joseph, S. Kadavath, J. Kernion, T. Conerly, S. El-Showk, N. Elhage, Z. Hatfield-Dodds, D. Hernandez, T. Hume, S. Johnston, S. Kravec, L. Lovitt, N. Nanda, C. Olsson, D. Amodei, T. B. Brown, J. Clark, S. McCandlish, C. Olah, B. Mann, and J. Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Baker Law. Getty images vs. stability ai: A landmark case in copyright and ai, 2023. URL https://www.bakerlaw.com/getty-images-v-stability-ai/.

R. J. N. Baldock, H. Maennel, and B. Neyshabur. Deep learning through the lens of example difficulty. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=WWRBHhH158K.

G. Bao, Y. Zhao, Z. Teng, L. Yang, and Y. Zhang. Fast-detectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Bpcgcr8E8Z.

A. Barbu, D. Mayo, J. Alverio, W. Luo, C. Wang, D. Gutfreund, J. Tenenbaum, and B. Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9453–9463, 2019.

G.-O. Barbulescu and P. Triantafillou. To each (textual sequence) its own: Improving memorized-data unlearning in large language models, 2024. URL https://arxiv.org/abs/2405.03097.

H. B. Barlow. Single units and sensation: A neuron doctrine for perceptual psychology. *Perception*, 1:371–394, 1972.

P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler. Benign overfitting in linear regression. *Proceedings of the National*

Academy of Sciences, 117(48):30063–30070, 2020.

N. Belrose, Z. Furman, L. Smith, D. Halawi, I. Ostrovsky, L. McKinney, S. Biderman, and J. Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023.

L. Ben Allal, A. Lozhkov, G. Penedo, T. Wolf, and L. von Werra. Cosmopedia, 2024. URL https://huggingface.co/datasets/HuggingFaceTB/cosmopedia.

Q. Bertrand, A. J. Bose, A. Duplessis, M. Jiralerspong, and G. Gidel. On the stability of iterative retraining of generative models on their own data. *arXiv preprint arXiv:2310.00429*, 2023.

J. Betker, G. Goh, L. Jing, T. Brooks, J. Wang, L. Li, L. Ouyang, J. Zhuang, J. Lee, Y. Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

J. Betley, D. Tan, N. Warncke, A. Sztyber-Betley, X. Bao, M. Soto, N. Labenz, and O. Evans. Emergent misalignment: Narrow finetuning can produce broadly misaligned llms. *arXiv preprint arXiv:2502.17424*, 2025.

K. Bhaila, M.-H. Van, and X. Wu. Soft prompting for unlearning in large language models. In L. Chiruzzo, A. Ritter, and L. Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4046–4056, Albuquerque, New Mexico, Apr. 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.204/.

F. Bianchi, M. Suzgun, G. Attanasio, P. Rottger, D. Jurafsky, T. Hashimoto, and J. Zou. Safety-tuned llamas: Lessons from improving the safety of large language models that follow instructions. *arXiv preprint arXiv:2309.07875*, 2023.

S. Biderman, H. Schoelkopf, Q. Anthony, H. Bradley, K. O'Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, A. Skowron, L. Sutawika, and O. Van Der Wal. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023a.

S. Biderman, H. Schoelkopf, Q. G. Anthony, H. Bradley, K. O'Brien, E. Hallahan, M. A. Khan, S. Purohit, U. S. Prashanth, E. Raff, A. Skowron, L. Sutawika, and O. van der Wal. Pythia: A suite for analyzing large language models across training and scaling. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 2397–2430. PMLR, 2023b.

URL https://proceedings.mlr.press/v202/biderman23a.html.

Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi. Piqa: Reasoning about physical commonsense in natural language. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020a.

Y. Bisk, R. Zellers, R. Le Bras, J. Gao, and Y. Choi. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of AAAI*, 2020b.

Y. Bitton, N. B. Guetta, R. Yosef, Y. Elovici, M. Bansal, G. Stanovsky, and R. Schwartz. Winogavil: Gamified association benchmark to challenge vision-and-language models, 2022.

L. Bossard, M. Guillaumin, and L. Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

M. Briesch, D. Sobania, and F. Rothlauf. Large language models suffer from their own output: An analysis of the self-consuming training loop. 2024. URL https://arxiv.org/abs/2311.16822.

J. Brokman, A. Giloni, O. Hofman, R. Vainshtein, H. Kojima, and G. Gilboa. Identifying memorization of diffusion models through p-laplace analysis. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 295–307. Springer, 2025.

M. B. Brown. 400: A method for combining non-independent, one-sided tests of significance. *Biometrics*, pages 987–992, 1975.

T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020a.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020b.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. Mc-

Candlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020c. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

N. Carlini, Ú. Erlingsson, and N. Papernot. Distribution density, tails, and outliers in machine learning: Metrics and applications. *ArXiv*, abs/1910.13427, 2019a.

N. Carlini, U. Erlingsson, and N. Papernot. Distribution density, tails, and outliers in machine learning: Metrics and applications. *arXiv preprint arXiv:1910.13427*, 2019b.

N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX security symposium (USENIX security 19)*, pages 267–284, 2019c.

N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer. Membership inference attacks from first principles. *arXiv preprint arXiv:2112.03570*, 2021a.

N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association, Aug. 2021b. ISBN 978-1-939133-24-3. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, A. Oprea, and C. Raffel. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*, 2021c.

N. Carlini, F. Tramèr, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, Ú. Erlingsson, A. Oprea, and C. Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association, Aug. 2021d. ISBN 978-1-939133-24-3. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022a.

N. Carlini, S. Chien, M. Nasr, S. Song, A. Terzis, and F. Tramèr. Membership inference attacks from first principles. In

2022 IEEE Symposium on Security and Privacy (SP), pages 1897–1914, 2022b. doi: 10.1109/SP46214.2022.9833649.

N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramer, B. Balle, D. Ippolito, and E. Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5253–5270, 2023a.

N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramèr, and C. Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023b. URL https://openreview.net/pdf?id=TatRHT_1cK.

N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, and C. Zhang. Quantifying memorization across neural language models, 2023c.

N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, and C. Zhang. Quantifying memorization across neural language models, 2023d. URL https://arxiv.org/abs/2202.07646.

P. Chao, A. Robey, E. Dobriban, H. Hassani, G. J. Pappas, and E. Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

P. Chao, E. Debenedetti, A. Robey, M. Andriushchenko, F. Croce, V. Sehwag, E. Dobriban, N. Flammarion, G. J. Pappas, F. Tramèr, et al. Jailbreakbench: An open robustness benchmark for jailbreaking large language models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

S. Chatterjee. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. *arXiv preprint arXiv:2002.10657*, 2020.

N. S. Chatterji and P. M. Long. Finite-sample analysis of interpolating linear classifiers in the overparameterized regime. *J. Mach. Learn. Res.*, 22:129–1, 2021.

R. Chavhan, D. Li, and T. Hospedales. Conceptprune: Concept editing in diffusion models via skilled neuron pruning. *arXiv preprint arXiv:2405.19237*, 2024.

C. Chen, D. Liu, M. Shah, and C. Xu. Exploring local memorization in diffusion models via bright ending attention. *arXiv preprint arXiv:2410.21665*, 2024a.

T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollar, and C. L. Zitnick. Microsoft coco captions: Data collection and evaluation server, 2015.

Z. Chen, J. Wang, W. Wang, G. Chen, E. Xie, P. Luo, and T. Lu. Fast: Faster arbitrarily-

shaped text detector with minimalist kernel representation, 2021.

Z. Chen, Z. Liu, T. Lin, Y. Jiang, and Q. Zhang. Towards memorization-free diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

G. Cheng, J. Han, and X. Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, Oct 2017. ISSN 1558-2256. doi: 10.1109/jproc.2017.2675998. URL http://dx.doi.org/10.1109/JPROC.2017.2675998.

M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, and J. Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

B. Cheung, A. Terekhov, Y. Chen, P. Agrawal, and B. Olshausen. Superposition of many models into one. *Advances in neural information processing systems*, 32, 2019.

W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, I. Stoica, and E. P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL https://lmsys.org/blog/2023-03-30-vicuna/.

K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In D. Wu, M. Carpuat, X. Carreras, and E. M. Vecchi, editors, *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL https://aclanthology.org/W14-4012.

M. Choi, D. Rim, D. Lee, and J. Choo. Opt-out: Investigating entity-level unlearning for large language models via optimal transport, 2024. URL https://arxiv.org/abs/2406.12329.

A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and

N. Fiedel. Palm: Scaling language modeling with pathways, 2022. URL https://arxiv.org/abs/2204.02311.

C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, 2019.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *ArXiv preprint*, abs/1803.05457, 2018a. URL https://arxiv.org/abs/1803.05457.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018b.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv:1803.05457v1*, 2018c.

A. Cloud, J. Goldman-Wetzler, E. Wybitul, J. Miller, and A. M. Turner. Gradient routing: Masking gradients to localize computation in neural networks, 2024. URL https://arxiv.org/abs/2410.04332.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, C. Hesse, and J. Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021a.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *ArXiv preprint*, abs/2110.14168, 2021b. URL https://arxiv.org/abs/2110.14168.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021c.

T. Computer. Redpajama: an open dataset for training large language models. 2023. URL https://github.com/togethercomputer/RedPajama-Data.

O. Contributors. Opencompass: A universal evaluation platform for foundation models, 2023.

A. F. Cooper, K. Lee, J. Grimmelmann, D. Ippolito, C. Callison-Burch, C. A. Choquette-Choo, N. Mireshghallah, M. Brundage, D. Mimno, M. Z. Choksi, et al. Report of the 1st workshop on generative ai and law. *arXiv preprint arXiv:2311.06477*, 2023.

D. Das, J. Zhang, and F. Tramèr. Blind baselines beat membership inference attacks

for foundation models, 2024. URL https://arxiv.org/abs/2406.16201.

Databricks. Introducing dbrx: A new state-of-the-art open llm. https://www.databricks.com/blog/introducing-dbrx-new-state-art-open-llm, 2025.

DatologyAI, A. Carranza, A. Deng, P. Maini, M. Razzak, J. Urbanek, A. Abbas, P. Burstein, N. Cao, P. Goyal, J. McGrath, F. Pan, J. Wills, H. Yin, V. Kada, V. Shah, V. Veerendranath, B. Gaza, A. Morcos, and M. Leavitt. DatologyAI Technical Deep-Dive: Image-Text Data Curation at the Billion-Sample Scale. Nov. 2024. URL https://blog.datologyai.com/technical-deep-dive-curating-our-way-to-a-state-of-the-art-text-dataset/.

L. Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer. QLoRA: Efficient finetuning of quantized llms. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

K. D. Dhole, V. Gangal, S. Gehrmann, A. Gupta, Z. Li, S. Mahamood, A. Mahendiran, S. Mille, A. Srivastava, S. Tan, T. Wu, J. Sohl-Dickstein, J. D. Choi, E. Hovy, O. Dusek, S. Ruder, S. Anand, N. Aneja, R. Banjade, L. Barthe, H. Behnke, I. Berlot-Attwell, C. Boyle, C. Brun, M. A. S. Cabezudo, S. Cahyawijaya, E. Chapuis, W. Che, M. Choudhary, C. Clauss, P. Colombo, F. Cornell, G. Dagan, M. Das, T. Dixit, T. Dopierre, P.-A. Dray, S. Dubey, T. Ekeinhor, M. D. Giovanni, R. Gupta, R. Gupta, L. Hamla, S. Han, F. Harel-Canada, A. Honore, I. Jindal, P. K. Joniak, D. Kleyko, V. Kovatchev, K. Krishna, A. Kumar, S. Langer, S. R. Lee, C. J. Levinson, H. Liang, K. Liang, Z. Liu, A. Lukyanenko, V. Marivate, G. de Melo, S. Meoni, M. Meyer, A. Mir, N. S. Moosavi, N. Muennighoff, T. S. H. Mun, K. Murray, M. Namysl, M. Obedkova, P. Oli, N. Pasricha, J. Pfister, R. Plant, V. Prabhu, V. Pais, L. Qin, S. Raji, P. K. Rajpoot, V. Raunak, R. Rinberg, N. Roberts, J. D. Rodriguez, C. Roux, V. P. H. S., A. B. Sai, R. M. Schmidt, T. Scialom, T. Sefara, S. N. Shamsi, X. Shen, H. Shi, Y. Shi, A. Shvets, N. Siegel, D. Sileo, J. Simon, C. Singh, R. Sitelew, P. Soni, T. Sorensen, W. Soto, A. Srivastava, K. A. Srivatsa, T. Sun, M. V. T, A. Tabassum, F. A. Tan, R. Teehan, M. Tiwari, M. Tolkiehn,

A. Wang, Z. Wang, G. Wang, Z. J. Wang, F. Wei, B. Wilie, G. I. Winata, X. Wu, W. Wydmański, T. Xie, U. Yaseen, M. Yee, J. Zhang, and Y. Zhang. Nl-augmenter: A framework for task-sensitive natural language augmentation, 2023.

C. Ding, J. Wu, Y. Yuan, J. Lu, K. Zhang, A. Su, X. Wang, and X. He. Unified parameter-efficient unlearning for LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=zONMuIVCAT.

E. Dohmatob, Y. Feng, P. Yang, F. Charton, and J. Kempe. A tale of tails: Model collapse as a change of scaling laws. *arXiv preprint arXiv:2402.07043*, 2024.

Y. R. Dong, H. Lin, M. Belkin, R. Huerta, and I. Vulić. UNDIAL: Self-distillation with adjusted logits for robust unlearning in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8827–8840, Albuquerque, New Mexico, Apr. 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.444/.

J. Doshi and A. C. Stickland. Does unlearning truly unlearn? A black box evaluation of LLM unlearning methods. *arXiv preprint arXiv:2411.12103*, 2024.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

H. Duan, A. Dziedzic, N. Papernot, and F. Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models. In *Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

M. Duan, A. Suri, N. Mireshghallah, S. Min, W. Shi, L. Zettlemoyer, Y. Tsvetkov, Y. Choi, D. Evans, and H. Hajishirzi. Do membership inference attacks work on large language models? In *First Conference on Language Modeling*, 2024a. URL https://openreview.net/forum?id=av0D19pSkU.

M. Duan, A. Suri, N. Mireshghallah, S. Min, W. Shi, L. Zettlemoyer, Y. Tsvetkov, Y. Choi, D. Evans, and H. Hajishirzi. Do membership inference attacks work on large language models? 2024b.

J. Dubiński, A. Kowalczuk, F. Boenisch, and A. Dziedzic. Cdi: Copyrighted data identification in diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18674–18684, 2025.

Y. Elazar, A. Bhagia, I. Magnusson, A. Ravichander, D. Schwenk, A. Suhr, E. P. Walsh, D. Groeneveld, L. Soldaini,

S. Singh, H. Hajishirzi, N. A. Smith, and J. Dodge. What's in my big data? In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=RvfPnOkPV4.

R. Eldan and Y. Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023a.

R. Eldan and Y. Li. Tinystories: How small can language models be and still speak coherent english?, 2023b. URL https://arxiv.org/abs/2305.07759.

R. Eldan and M. Russinovich. Who's harry potter? approximate unlearning in llms. *arXiv preprint arXiv:2310.02238*, 2023.

M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

C. Fan, J. Liu, L. Lin, J. Jia, R. Zhang, S. Mei, and S. Liu. Simplicity prevails: Rethinking negative preference optimization for LLM unlearning. In *Neurips Safe Generative AI Workshop 2024*, 2024. URL https://openreview.net/forum?id=pVACX02m0p.

C. Fan, J. Jia, Y. Zhang, A. Ramakrishna, M. Hong, and S. Liu. Towards llm unlearning resilient to relearning attacks: A sharpness-aware minimization perspective and beyond. *arXiv preprint arXiv:2502.05374*, 2025.

L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Pattern Recognition Workshop*, 2004.

V. Feldman. Does learning require memorization? a short tale about a long tail. *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, 2020a.

V. Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020b.

V. Feldman and C. Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33:2881–2891, 2020.

J. Frankle, D. J. Schwab, and A. S. Morcos. The early phase of neural network training. In *International Conference on Learning Representations*, 2020a. URL https://openreview.net/forum?id=Hkl1iRNFwS.

J. Frankle, D. J. Schwab, and A. S. Morcos. The early phase of neural network training. *arXiv preprint arXiv:2002.10365*, 2020b.

R. Futrell, K. Mahowald, and E. Gibson. Large-scale evidence of dependency length minimization in 37 languages. *Proceedings of the National Academy of Sciences*, 112(33):10336–10341, 2015.

S. Y. Gadre, G. Ilharco, A. Fang, J. Hayase, G. Smyrnis, T. Nguyen, R. Marten, M. Wortsman, D. Ghosh, J. Zhang, E. Orgad, R. Entezari, G. Daras, S. Pratt, V. Ramanujan, Y. Bitton, K. Marathe, S. Mussmann, R. Vencu, M. Cherti, R. Krishna, P. W. Koh, O. Saukh, A. Ratner, S. Song, H. Hajishirzi, A. Farhadi, R. Beaumont, S. Oh, A. Dimakis, J. Jitsev, Y. Carmon, V. Shankar, and L. Schmidt. Datacomp: In search of the next generation of multimodal datasets, 2023.

J.-l. Gailly and M. Adler. zlib compression library. 2004. URL http://www.dspace.cam.ac.uk/handle/1810/3486.

R. Gandikota, S. Feucht, S. Marks, and D. Bau. Erasing conceptual knowledge from language models. *arXiv preprint arXiv:2410.02760*, 2024.

C. Gao, L. Wang, C. Weng, X. Wang, and Q. Zhu. Practical unlearning for large language models. *arXiv preprint arXiv:2407.10223*, 2024a.

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

L. Gao, Y. Niu, T. Tang, S. Avestimehr, and M. Annavaram. Ethos: Rectifying language models in orthogonal parameter space. *arXiv preprint arXiv:2403.08994*, 2024b.

L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, and A. Zou. The Language Model Evaluation Harness, 07 2024c. URL https://zenodo.org/records/12608602.

T. Gao, X. Yao, and D. Chen. Simcse: Simple contrastive learning of sentence embeddings. In *2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021*, pages 6894–6910. Association for Computational Linguistics (ACL), 2021.

S. Garg, S. Balakrishnan, Z. Kolter, and Z. Lipton. RATT: Leveraging unlabeled data to guarantee generalization. *arXiv preprint arXiv:2105.00303*, 2021. URL https://arxiv.org/abs/2105.00303.

J. Geiping, A. Stein, M. Shu, K. Saifullah, Y. Wen, and T. Goldstein. Coercing llms to do and reveal (almost) anything. *arXiv preprint arXiv:2402.14020*, 2024.

Gemma Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, et al.

Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

M. Geva, R. Schuster, J. Berant, and O. Levy. Transformer feed-forward layers are key-value memories. In M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

E. Gibson et al. The dependency locality theory: A distance-based theory of linguistic complexity. *Image, language, brain*, 2000:95–126, 2000.

S. Goel, A. Prabhu, A. Sanyal, S.-N. Lim, P. Torr, and P. Kumaraguru. Towards adversarial evaluations for inexact machine unlearning. *arXiv preprint arXiv:2201.06640*, 2022.

A. Golatkar, A. Achille, and S. Soatto. Forgetting outside the box: Scrubbing deep networks of information accessible from input-output observations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pages 383–398. Springer, 2020.

S. Golchin and M. Surdeanu. Time travel in llms: Tracing data contamination in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=2Rwq6c3tvr.

Google DeepMind. Gemini. https://gemini.google.com/, 2025.

S. Goyal, P. Maini, Z. C. Lipton, A. Raghunathan, and J. Z. Kolter. Scaling Laws for Data Filtering—Data Curation cannot be Compute Agnostic. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

M. M. Grynbaum and R. Mac. The times sues openai and microsoft over a.i. use of copyrighted work https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html. 2023a. URL https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html.

M. M. Grynbaum and R. Mac. Sarah silverman and authors sue openai and meta over copyright infringement. 2023b. URL https:

//www.nytimes.com/2023/07/10/arts/sarah-silverman-lawsuit-openai-meta.html.

T. Gu, K. Huang, R. Luo, Y. Yao, Y. Yang, Y. Teng, and Y. Wang. Meow: Memory supervised llm unlearning via inverted facts, 2024. URL https://arxiv.org/abs/2409.11844.

Y. Gu, O. Tafjord, B. Kuehl, D. Haddad, J. Dodge, and H. Hajishirzi. Olmes: A standard for language model evaluations. 2025. URL https://arxiv.org/abs/2406.08446.

S. Gugger, L. Debut, T. Wolf, P. Schmid, Z. Mueller, S. Mangrulkar, M. Sun, and B. Bossan. Accelerate: Training and inference at scale made simple, efficient and adaptable. https://github.com/huggingface/accelerate, 2022.

S. Gunasekar, Y. Zhang, J. Aneja, C. C. T. Mendes, A. Del Giorno, S. Gopi, M. Javaheripi, P. Kauffmann, G. de Rosa, O. Saarikivi, et al. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*, 2023.

C. Guo, T. Goldstein, A. Hannun, and L. Van Der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Y. Guo, G. Shang, M. Vazirgiannis, and C. Clavel. The curious decline of linguistic diversity: Training language models on synthetic text. 2024. URL https://arxiv.org/abs/2311.09807.

N. Habib, C. Fourrier, H. Kydlíček, T. Wolf, and L. Tunstall. Lighteval: A lightweight framework for llm evaluation. 2023. URL https://github.com/huggingface/lighteval.

G. Hacohen, L. Choshen, and D. Weinshall. Let's agree to agree: Neural networks share classification order on real datasets. In *International Conference on Machine Learning*, pages 3950–3960. PMLR, 2020.

N. Haim, G. Vardi, G. Yehudai, O. Shamir, and M. Irani. Reconstructing training data from trained neural networks. *arXiv preprint arXiv:2206.07758*, 2022.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

L. He, M. Xia, and P. Henderson. What is in your safe data? identifying benign data that breaks safety, 2024. URL https://arxiv.org/abs/2404.01099.

W. Heilbut, D. Maggiacomo, and M. Casey. Image-generative ai: Has technology finally evolved beyond the parameters of the modern day fair use defense? URL https://www.barclaydamon.com/webfiles/BrightIdeas-2024-vol-33-no-1_WEB.pdf.

D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial

examples. *arXiv preprint arXiv:1907.07174*, 2019.

D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding, 2021b. URL https://arxiv.org/abs/2009.03300.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2021c.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021d.

M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

D. Hintersdorf, L. Struppek, K. Kersting, A. Dziedzic, and F. Boenisch. Finding nemo: Localizing neurons responsible for memorization in diffusion models. *Advances in Neural Information Processing Systems*, 37:88236–88278, 2024.

J. Ho and T. Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33: 6840–6851, 2020.

J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. d. L. Casas, L. A. Hendricks, J. Welbl, A. Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

S. Hooker, A. Courville, G. Clark, Y. Dauphin, and A. Frome. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.

J. Howard. Imagenette. URL https://github.com/fastai/imagenette/.

S. Hu, Y. Fu, S. Wu, and V. Smith. Unlearning or obfuscating? Jogging the memory of unlearned LLMs via benign relearning. In *The*

Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=fMNRYBvcQN.

X. Hu, D. Li, Z. Zheng, Z. Liu, B. Hu, and M. Zhang. Separate the wheat from the chaff: Model deficiency unlearning via parameter-efficient module operation, 2023.

B. Hui, J. Yang, Z. Cui, J. Yang, D. Liu, L. Zhang, T. Liu, J. Zhang, B. Yu, K. Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

B. Y. Idrissi, M. Arjovsky, M. Pezeshki, and D. Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. *arXiv preprint arXiv:2110.14503*, 2021.

G. Ilharco, M. Wortsman, R. Wightman, C. Gordon, N. Carlini, R. Taori, A. Dave, V. Shankar, H. Namkoong, J. Miller, H. Hajishirzi, A. Farhadi, and L. Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

S. A. Jacobs, M. Tanaka, C. Zhang, M. Zhang, L. Song, S. Rajbhandari, and Y. He. DeepSpeed Ulysses: System optimizations for enabling training of extreme long sequence transformer models. *arXiv preprint arXiv:2309.14509*, 2023. URL http://arxiv.org/abs/2309.14509.

A. Jacovi, A. Caciularu, O. Goldman, and Y. Goldberg. Stop uploading test data in plain text: Practical strategies for mitigating data contamination by evaluation benchmarks. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5075–5084, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.308. URL https://aclanthology.org/2023.emnlp-main.308.

N. Jain, A. Shrivastava, C. Zhu, D. Liu, A. Samuel, A. Panda, A. Kumar, M. Goldblum, and T. Goldstein. Refusal tokens: A simple way to calibrate refusals in large language models, 2024. URL https://arxiv.org/abs/2412.06748.

D. Jeon, D. Kim, and A. No. Understanding and mitigating memorization in generative models via sharpness of probability landscapes. In *Forty-second International Conference on Machine Learning*, 2025.

C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

J. Jia, Y. Zhang, Y. Zhang, J. Liu, B. Runwal, J. Diffenderfer, B. Kailkhura, and S. Liu. Soul: Unlocking the power of second-order optimization for LLM unlearning. *arXiv preprint arXiv:2404.18239*, 2024.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *ArXiv preprint*, abs/2310.06825, 2023a. URL https://arxiv.org/abs/2310.06825.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b. 2023b. URL https://arxiv.org/abs/2310.06825.

A. Q. Jiang, A. Sablayrolles, A. Roux, A. Mensch, B. Savary, C. Bamford, D. S. Chaplot, D. de las Casas, E. B. Hanna, F. Bressand, G. Lengyel, G. Bour, G. Lample, L. R. Lavaud, L. Saulnier, M.-A. Lachaux, P. Stock, S. Subramanian, S. Yang, S. Antoniak, T. L. Scao, T. Gervet, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mixtral of experts. 2024. URL https://arxiv.org/abs/2401.04088.

Y. Jiang, H. Lin, Y. Bai, B. Peng, Z. Liu, Y. Lyu, Y. Yang, Xingzheng, and J. Dong. Image-level memorization detection via inversion-based inference perturbation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=vwOq7twk7L.

Z. Jiang, C. Zhang, K. Talwar, and M. C. Mozer. Characterizing structural regularities of labeled data in overparameterized models. *arXiv preprint arXiv:2002.03206*, 2020.

Z. Jiang, C. Zhang, K. Talwar, and M. C. Mozer. Characterizing structural regularities of labeled data in overparameterized models. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5034–5044. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/jiang21k.html.

Q. Jin, B. Dhingra, Z. Liu, W. Cohen, and X. Lu. Pubmedqa: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, 2019.

Z. Jin, P. Cao, C. Wang, Z. He, H. Yuan, J. Li, Y. Chen, K. Liu, and J. Zhao. RWKU: Benchmarking real-world knowledge unlearning for large language models. *arXiv preprint arXiv:2406.10890*, 2024.

M. G. Johannes Welbl, Nelson F. Liu. Crowdsourcing multiple choice science questions. 2017.

E. Jones, H. Palangi, C. S. Ribeiro, V. Chandrasekaran, S. Mukherjee, A. Mitra, A. H. Awadallah, and E. Kamar. Teaching language models to hallucinate less with synthetic tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=xpw7V0P136.

M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In R. Barzilay and M.-Y. Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models, 2020.

A. Karamolegkou, J. Li, L. Zhou, and A. Søgaard. Copyright violations and large language models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=YokfK5VOoz.

T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

C. Kim and Y. Qi. A comprehensive survey on concept erasure in text-to-image diffusion models. *arXiv preprint arXiv:2502.14896*, 2025.

Y. Kim, S. Cha, and D. Kim. Are we truly forgetting? A critical re-examination of machine unlearning evaluation protocols. *arXiv preprint arXiv:2503.06991*, 2025.

KimiTeam, Y. Bai, Y. Bao, G. Chen, J. Chen, N. Chen, R. Chen, Y. Chen, Y. Chen, Y. Chen, Z. Chen, J. Cui, H. Ding, M. Dong, A. Du, C. Du, D. Du, Y. Du, Y. Fan, Y. Feng, K. Fu, B. Gao, H. Gao, P. Gao, T. Gao, X. Gu, L. Guan, H. Guo, J. Guo, H. Hu, X. Hao, T. He, W. He, W. He, C. Hong, Y. Hu, Z. Hu, W. Huang, Z. Huang, Z. Huang, T. Jiang, Z. Jiang, X. Jin, Y. Kang, G. Lai, C. Li, F. Li, H. Li, M. Li, W. Li, Y. Li, Y. Li, Z. Li, Z. Li, H. Lin, X. Lin, Z. Lin, C. Liu, C. Liu, H. Liu, J. Liu, J. Liu, L. Liu, S. Liu, T. Y. Liu, T. Liu, W. Liu, Y. Liu, Y. Liu, Y. Liu, Y. Liu, Z. Liu, E. Lu, L. Lu, S. Ma, X. Ma, Y. Ma, S. Mao, J. Mei, X. Men, Y. Miao, S. Pan, Y. Peng, R. Qin, B. Qu, Z. Shang, L. Shi, S. Shi, F. Song, J. Su, Z. Su, X. Sun, F. Sung, H. Tang, J. Tao, Q. Teng, C. Wang, D. Wang, F. Wang, H. Wang, J. Wang, J. Wang, J. Wang, S. Wang, S. Wang, Y. Wang, Y. Wang, Y. Wang, Y. Wang, Y. Wang, Z. Wang, Z. Wang, Z. Wang, C. Wei, Q. Wei, W. Wu, X. Wu, Y. Wu, C. Xiao, X. Xie, W. Xiong, B. Xu, J. Xu, J. Xu, L. H. Xu, L. Xu, S. Xu, W. Xu, X. Xu, Y. Xu, Z. Xu, J. Yan, Y. Yan, X. Yang, Y. Yang, Z. Yang, Z. Yang, Z. Yang, H. Yao, X. Yao,

W. Ye, Z. Ye, B. Yin, L. Yu, E. Yuan, H. Yuan, M. Yuan, H. Zhan, D. Zhang, H. Zhang, W. Zhang, X. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, Y. Zhang, Z. Zhang, H. Zhao, Y. Zhao, H. Zheng, S. Zheng, J. Zhou, X. Zhou, Z. Zhou, Z. Zhu, W. Zhuang, and X. Zu. Kimi k2: Open agentic intelligence. 2025. URL https://arxiv.org/abs/2507.20534.

D. Kingma, T. Salimans, B. Poole, and J. Ho. Variational diffusion models. *Advances in neural information processing systems*, 34: 21696–21707, 2021.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein. A watermark for large language models. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, editors, *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 17061–17084. PMLR, 2023. URL https://proceedings.mlr.press/v202/kirchenbauer23a.html.

J. Kirchenbauer, J. Geiping, Y. Wen, M. Shu, K. Saifullah, K. Kong, K. Fernando, A. Saha, M. Goldblum, and T. Goldstein. On the reliability of watermarks for large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=DEJIDCmWOz.

A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby. Big transfer (bit): General visual representation learning. In *European conference on computer vision*, pages 491–507. Springer, 2020.

T. Korbak, K. Shi, A. Chen, R. Bhalerao, C. L. Buckley, J. Phang, S. Bowman, and E. Perez. Pretraining language models with human preferences. *arXiv preprint arXiv:2302.08582*, 2023.

J. T. Kost and M. P. McDermott. Combining dependent p-values. *Statistics & Probability Letters*, 60(2):183–190, 2002.

J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.

K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.

cc/paper_files/paper/2023/hash/575c450013d0e99e4b0ecf82bd1afaa4-Abstract-Conference.html.

A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Computer Science Department, University of Toronto, 2009.

A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

M. Kurmanji, P. Triantafillou, and E. Triantafillou. The brainy student: Scalable unlearning by selectively disobeying the teacher, 2023. URL https://openreview.net/forum?id=f9eHl5mKx5i.

W. Kwon, Z. Li, S. Zhuang, Y. Sheng, L. Zheng, C. H. Yu, J. E. Gonzalez, H. Zhang, and I. Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy. Race: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*, 2017.

G. K. R. Lau, X. Niu, H. Dao, J. Chen, C.-S. Foo, and B. K. H. Low. Waterfall: Framework for robust and scalable text watermarking and provenance for llms, 2024. URL https://arxiv.org/abs/2407.04411.

W. Laurito, B. Davis, P. Grietzer, T. Gavenčiak, A. Böhm, and J. Kulveit. Ai ai bias: Large language models favor their own generated content. *ArXiv preprint*, abs/2407.12856, 2024. URL https://arxiv.org/abs/2407.12856.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

K. Lee, D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-Burch, and N. Carlini. Deduplicating training data makes language models better. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8424–8445, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.577. URL https://aclanthology.org/2022.acl-long.577.

J. Li, D. Li, S. Savarese, and S. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023a.

J. Li, A. Fang, G. Smyrnis, M. Ivgi, M. Jordan, S. Gadre, H. Bansal, E. Guha, S. Keh, K. Arora, S. Garg, R. Xin, N. Muennighoff, R. Heckel, J. Mercat, M. Chen,

S. Gururangan, M. Wortsman, A. Albalak, Y. Bitton, M. Nezhurina, A. Abbas, C.-Y. Hsieh, D. Ghosh, J. Gardner, M. Kilian, H. Zhang, R. Shao, S. Pratt, S. Sanyal, G. Ilharco, G. Daras, K. Marathe, A. Gokaslan, J. Zhang, K. Chandu, T. Nguyen, I. Vasiljevic, S. Kakade, S. Song, S. Sanghavi, F. Faghri, S. Oh, L. Zettlemoyer, K. Lo, A. El-Nouby, H. Pouransari, A. Toshev, S. Wang, D. Groeneveld, L. Soldaini, P. W. Koh, J. Jitsev, T. Kollar, A. G. Dimakis, Y. Carmon, A. Dave, L. Schmidt, and V. Shankar. Datacomp-lm: In search of the next generation of training sets for language models. *arXiv preprint arXiv:2406.11794*, 2024a.

M. Li, M. Soltanolkotabi, and S. Oymak. Gradient descent with early stopping is provably robust to label noise for overparameterized neural networks. In S. Chiappa and R. Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 4313–4324. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/li20j.html.

N. Li, A. Pan, A. Gopal, S. Yue, D. Berrios, A. Gatti, J. D. Li, A.-K. Dombrowski, S. Goel, L. Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024b.

R. Li, L. B. Allal, Y. Zi, N. Muennighoff, D. Kocetkov, C. Mou, M. Marone, C. Akiki, J. Li, J. Chim, Q. Liu, E. Zheltonozhskii, T. Y. Zhuo, T. Wang, O. Dehaene, M. Davaadorj, J. Lamy-Poirier, J. Monteiro, O. Shliazhko, N. Gontier, N. Meade, A. Zebaze, M.-H. Yee, L. K. Umapathi, J. Zhu, B. Lipkin, M. Oblokulov, Z. Wang, R. Murthy, J. Stillerman, S. S. Patel, D. Abulkhanov, M. Zocca, M. Dey, Z. Zhang, N. Fahmy, U. Bhattacharyya, W. Yu, S. Singh, S. Luccioni, P. Villegas, M. Kunakov, F. Zhdanov, M. Romero, T. Lee, N. Timor, J. Ding, C. Schlesinger, H. Schoelkopf, J. Ebert, T. Dao, M. Mishra, A. Gu, J. Robinson, C. J. Anderson, B. Dolan-Gavitt, D. Contractor, S. Reddy, D. Fried, D. Bahdanau, Y. Jernite, C. M. Ferrandis, S. Hughes, T. Wolf, A. Guha, L. von Werra, and H. de Vries. Starcoder: may the source be with you! 2023b.

X. Li, P. Yu, C. Zhou, T. Schick, L. Zettlemoyer, O. Levy, J. Weston, and M. Lewis. Self-alignment with instruction backtranslation. *arXiv preprint arXiv:2308.06259*, 2023c.

X. Li, H. Tu, M. Hui, Z. Wang, B. Zhao, J. Xiao, S. Ren, J. Mei, Q. Liu, H. Zheng, et al. What if we recaption billions of web images with llama-3? *arXiv preprint arXiv:2406.08478*, 2024c.

X. Li, P. Yu, C. Zhou, T. Schick, O. Levy, L. Zettlemoyer, J. Weston, and M. Lewis. Self-alignment with instruction backtranslation. 2024d. URL https://arxiv.org/abs/2308.06259.

Y. Li, S. Bubeck, R. Eldan, A. Del Giorno,

S. Gunasekar, and Y. T. Lee. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*, 2023d.

Y. Li, S. Bubeck, R. Eldan, A. Del Giorno, S. Gunasekar, and Y. T. Lee. Textbooks are all you need ii: **phi-1.5** technical report. *arXiv preprint arXiv:2309.05463*, 2023e.

Y. Li, S. Bubeck, R. Eldan, A. D. Giorno, S. Gunasekar, and Y. T. Lee. Textbooks Are All You Need II: Phi-1.5 technical report, 2023f. URL https://arxiv.org/abs/2309.05463.

P. Liang. Open weight llms. https://x.com/percyliang/status/1800650546208035123, 2024.

C.-Y. Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004a. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004b.

S. Lin, J. Hilton, and O. Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2021.

B. Liu, Q. Liu, and P. Stone. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, pages 243–254. PMLR, 2022a.

H. Liu, J. Liu, L. Cui, Z. Teng, N. Duan, M. Zhou, and Y. Zhang. Logiqa 2.0 — an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, pages 1–16, 2023a. doi: 10.1109/TASLP.2023.3293046.

J. Liu, S. Min, L. Zettlemoyer, Y. Choi, and H. Hajishirzi. Infini-gram: Scaling unbounded n-gram language models to a trillion tokens. *arXiv preprint arXiv:2401.17377*, 2024a.

S. Liu, J. Niles-Weed, N. Razavian, and C. Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020.

S. Liu, Z. Zhu, Q. Qu, and C. You. Robust training under label noise by over-parameterization. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14153–14172. PMLR, 17–23 Jul 2022b.

S. Liu, Y. Yao, J. Jia, S. Casper, N. Baracaldo, P. Hase, X. Xu, Y. Yao, H. Li, K. R. Varshney, et al. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*, 2024b.

X. Liu, L. Hu, P. Bailis, I. Stoica, Z. Deng, A. Cheung, and H. Zhang. Online

speculative decoding. *arXiv preprint arXiv:2310.07177*, 2023b.

Y. Liu, H. Hu, X. Chen, X. Zhang, and L. Sun. Watermarking text data on large language models for dataset copyright. *ArXiv preprint*, abs/2305.13257, 2023c. URL https://arxiv.org/abs/2305.13257.

Y. Liu, D. Iter, Y. Xu, S. Wang, R. Xu, and C. Zhu. G-eval: NLG evaluation using gpt-4 with better human alignment. In H. Bouamor, J. Pino, and K. Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore, 2023d. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.153. URL https://aclanthology.org/2023.emnlp-main.153.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *ICCV*, pages 3730–3738. IEEE Computer Society, 2015. ISBN 978-1-4673-8391-2. URL http://dblp.uni-trier.de/db/conf/iccv/iccv2015.html#LiuLWT15.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

X. Lu, S. Welleck, J. Hessel, L. Jiang, L. Qin, P. West, P. Ammanabrolu, and Y. Choi. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609, 2022.

Y. Lu, Y. Li, Z. Wang, Q. Li, H. Zhang, and H. Yang. MACE: Mass concept erasure in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024a.

Z. Lu, A. Zhou, H. Ren, K. Wang, W. Shi, J. Pan, M. Zhan, and H. Li. MathGenie: Generating synthetic data with question back-translation for enhancing mathematical reasoning of LLMs. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2732–2747, Bangkok, Thailand, Aug. 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.151. URL https://aclanthology.org/2024.acl-long.151/.

J. Łucki, B. Wei, Y. Huang, P. Henderson, F. Tramèr, and J. Rando. An adversarial perspective on machine unlearning for AI safety. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=J5IRyTKZ9s.

R. Luukkonen, V. Komulainen, J. Luoma, A. Eskelinen, J. Kanerva, H.-M. Kupari,

F. Ginter, V. Laippala, N. Muennighoff, A. Piktus, et al. Fingpt: Large generative models for a small language. *arXiv preprint arXiv:2311.05640*, 2023.

A. Lynch, P. Guo, A. Ewart, S. Casper, and D. Hadfield-Menell. Eight methods to evaluate robust unlearning in LLMs. *arXiv preprint arXiv:2402.16835*, 2024.

Z. Ma, Q. Li, X. Zhang, T. Du, R. Lin, Z. Wang, S. Ji, and W. Chen. An inversion-based measure of memorization for diffusion models. *arXiv preprint arXiv:2405.05846*, 2024.

I. Magar and R. Schwartz. Data contamination: From memorization to exploitation. In S. Muresan, P. Nakov, and A. Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 157–165, Dublin, Ireland, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-short.18. URL https://aclanthology.org/2022.acl-short.18.

P. Maini. Phi-1.5 model: A case of comparing apples to oranges? 2023. URL https://pratyushmaini.github.io/phi-1_5/.

P. Maini, M. Yaghini, and N. Papernot. Dataset Inference: Ownership Resolution in Machine Learning. In *International Conference on Learning Representations (ICLR)*, 2021.

P. Maini, S. Garg, Z. C. Lipton, and J. Z. Kolter. Characterizing datapoints via second-split forgetting. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=yKDKNzjHg8N.

P. Maini, M. C. Mozer, H. Sedghi, Z. C. Lipton, J. Z. Kolter, and C. Zhang. Can neural network memorization be localized?, 2023. URL https://arxiv.org/abs/2307.09542.

P. Maini, Z. Feng, A. Schwarzschild, Z. Lipton, and Z. Kolter. TOFU: A Task of Fictitious Unlearning for LLMs. In *Conference on Language Modeling (COLM)*, 2024a.

P. Maini, H. Jia, N. Papernot, and A. Dziedzic. LLM Dataset Inference: Did you train on my dataset? In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024b.

P. Maini, H. Jia, N. Papernot, and A. Dziedzic. LLM dataset inference: Did you train on my dataset? In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024c. URL http://papers.nips.cc/paper_files/paper/2024/hash/e01519b47118e2f51aa643151350c905-Abstract-Conf
html.

P. Maini, S. Seto, H. Bai, D. Grangier, Y. Zhang, and N. Jaitly. Rephrasing the Web: A Recipe for Compute & Data-Efficient Language Modeling. In *Association for Computational Linguistics (ACL)*, 2024d.

P. Maini, S. Seto, R. Bai, D. Grangier, Y. Zhang, and N. Jaitly. Rephrasing the web: A recipe for compute and data-efficient language modeling. In L.-W. Ku, A. Martins, and V. Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14044–14072, Bangkok, Thailand, Aug. 2024e. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.757. URL https://aclanthology.org/2024.acl-long.757/.

P. Maini, S. Goyal, D. Sam, A. Robey, Y. Savani, Y. Jiang, A. Zou, Z. C. Lipton, and J. Z. Kolter. Safety pretraining: Toward the next generation of safe ai. *arXiv preprint arXiv:2504.16980*, 2025.

N. Maloyan, E. Verma, B. Nutfullin, and B. Ashinov. Trojan detection in large language models: Insights from the trojan detection challenge. *arXiv preprint arXiv:2404.13660*, 2024.

K. Mangalam and V. U. Prabhu. Do deep neural networks learn shallow learnable examples first? In *ICML 2019 Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019. URL https://openreview.net/forum?id=HkxHv4rn24.

G. Martínez, L. Watson, P. Reviriego, J. A. Hernández, M. Juarez, and R. Sarkar. Combining generative artificial intelligence (ai) and the internet: Heading towards evolution or degradation? *arXiv preprint arXiv:2303.01255*, 2023a.

G. Martínez, L. Watson, P. Reviriego, J. A. Hernández, M. Juarez, and R. Sarkar. Towards understanding the interplay of generative artificial intelligence and the internet. In *International Workshop on Epistemic Uncertainty in Artificial Intelligence*, pages 59–73. Springer, 2023b.

C. Marx, F. Calmon, and B. Ustun. Predictive multiplicity in classification. In *International Conference on Machine Learning*, pages 6765–6774. PMLR, 2020.

J. Mattern, F. Mireshghallah, Z. Jin, B. Schoelkopf, M. Sachan, and T. Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. In A. Rogers, J. Boyd-Graber, and N. Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11330–11343, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.719. URL https://aclanthology.org/2023.findings-acl.719.

M. Mazeika, L. Phan, X. Yin, A. Zou, Z. Wang, N. Mu, E. Sakhaee, N. Li, S. Basart,

B. Li, D. Forsyth, and D. Hendrycks. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. *arXiv preprint arXiv:2402.04249*, 2024.

M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

M. Meeus, I. Shilov, M. Faysse, and Y. de Montjoye. Copyright traps for large language models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024a. URL https://openreview.net/forum?id=LDq1JPdc55.

M. Meeus, I. Shilov, S. Jain, M. Faysse, M. Rei, and Y.-A. de Montjoye. Sok: Membership inference attacks on llms are rushing nowhere (and how to fix it). 2024b. URL https://api.semanticscholar.org/CorpusID:270737841.

A. Mekala, V. Dorna, S. Dubey, A. Lalwani, D. Koleczek, M. Rungta, S. Hasan, and E. Lobo. Alternate preference optimization for unlearning factual knowledge in large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 3732–3752, Abu Dhabi, UAE, Jan. 2025. Association for Computational Linguistics. URL https://aclanthology.org/2025.coling-main.252/.

K. Meng, D. Bau, A. Andonian, and Y. Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

X.-L. Meng. Posterior predictive p-values. *The annals of statistics*, 22(3):1142–1160, 1994.

Meta AI. Introducing llama 3: The most capable openly available llm to date. https://ai.meta.com/blog/meta-llama-3/, 2025.

C. Metz and K. Robertson. Openai seeks to dismiss parts of the new york times's lawsuit. *The New York Times*, 2024. URL https://www.nytimes.com/2024/02/27/technology/openai-new-york-times-lawsuit.html.

T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*, 2018a.

T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of EMNLP*, 2018b.

S. Min, S. Gururangan, E. Wallace, H. Hajishirzi, N. A. Smith, and L. Zettlemoyer. Silo language models: Isolating legal risk in a nonparametric datastore. *arXiv preprint arXiv:2308.04430*, 2023.

S. Mindermann, J. M. Brauner, M. T. Razzak, M. Sharma, A. Kirsch, W. Xu, B. Höltgen, A. N. Gomez, A. Morisot, S. Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pages 15630–15649. PMLR, 2022.

A. Mishra, S. Rahman, H. J. Kim, K. Mitra, and E. R. Hruschka. Characterizing large language models as rationalizers of knowledge-intensive tasks. *ArXiv preprint*, abs/2311.05085, 2023. URL https://arxiv.org/abs/2311.05085.

E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn. Detectgpt: zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

N. Muennighoff, A. M. Rush, B. Barak, T. L. Scao, A. Piktus, N. Tazi, S. Pyysalo, T. Wolf, and C. Raffel. Scaling data-constrained language models. *arXiv preprint arXiv:2305.16264*, 2023.

A. Muresanu, A. Thudi, M. R. Zhang, and N. Papernot. Unlearnable algorithms for in-context learning, 2024. URL https://arxiv.org/abs/2402.00751.

P. Nakkiran, G. Kaplun, Y. Bansal, T. Yang, B. Barak, and I. Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

N. Nanda, S. Rajamanoharan, J. Kramar, and R. Shah. Fact finding: Attempting to reverse-engineer factual recall on the neuron level, Dec 2023. URL https://www.alignmentforum.org/posts/iGuwZTHWb6DFY3sKB/fact-finding-attempting-to-reverse-engineer-fa

S. Narasimhaswamy, U. Bhattacharya, X. Chen, I. Dasgupta, S. Mitra, and M. Hoai. Handiffuser: Text-to-image generation with realistic hand appearances. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

A. Narayanan and S. Kapoor. Ai snake oil., 2023. URL https://www.aisnakeoil.com/. Newsletter.

M. Nasr, N. Carlini, J. Hayase, M. Jagielski, A. F. Cooper, D. Ippolito, C. A. Choquette-Choo, E. Wallace, F. Tramèr, and K. Lee. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035*, 2023a.

M. Nasr, N. Carlini, J. Hayase, M. Jagielski, A. F. Cooper, D. Ippolito, C. A. Choquette-Choo, E. Wallace, F. Tramèr, and K. Lee. Scalable extraction of training data from (production) language models, 2023b. URL https://arxiv.org/abs/2311.17035.

Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

NewYorkTimes. The times sues openai and microsoft over a.i. use of copyrighted work. https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html, 2023.

T. Nguyen, S. Y. Gadre, G. Ilharco, S. Oh, and L. Schmidt. Improving multimodal datasets with image captioning, 2023.

T. Nguyen, Y. Li, O. Golovneva, L. Zettlemoyer, S. Oh, L. Schmidt, and X. Li. Recycling the web: A method to enhance pre-training data quality and quantity for language models. 2025. URL https://arxiv.org/abs/2506.04689.

T. T. Nguyen, T. T. Huynh, P. L. Nguyen, A. W.-C. Liew, H. Yin, and Q. V. H. Nguyen. A survey of machine unlearning. *arXiv preprint arXiv:2209.02299*, 2022.

A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021.

M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

M. Ning, E. Sangineto, A. Porrello, S. Calderara, and R. Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning*, pages 26245–26265. PMLR, 2023.

C. OAG. Ccpa regulations: Final regulation text. *Office of the Attorney General, California Department of Justice*, 2021.

T. OLMo, P. Walsh, L. Soldaini, D. Groeneveld, K. Lo, S. Arora, A. Bhagia, Y. Gu, S. Huang, M. Jordan, N. Lambert, D. Schwenk, O. Tafjord, T. Anderson, D. Atkinson, F. Brahman, C. Clark, P. Dasigi, N. Dziri, M. Guerquin, H. Ivison, P. W. Koh, J. Liu, S. Malik, W. Merrill, L. J. V. Miranda, J. Morrison, T. Murray, C. Nam, V. Pyatkin, A. Rangapur, M. Schmitz, S. Skjonsberg, D. Wadden, C. Wilhelm, M. Wilson, L. Zettlemoyer, A. Farhadi, N. A. Smith, and H. Hajishirzi. 2 olmo 2 furious. 2024. URL https://arxiv.org/abs/2501.00656.

OpenAI. Gpt-4 technical report, 2023. URL https://arxiv.org/abs/2303.08774.

OpenAI. Openai. https://openai.com/, 2025.

OpenAI. Introducting gpt-5. 2025. URL https://www.youtube.com/live/0Uu_VJeVVfo?feature=shared&t=1982.

Y. Oren, N. Meister, N. S. Chatterji, F. Ladhak, and T. Hashimoto. Proving test set contamination in black-box language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=KS8mIvetg2.

L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

M. Oya. Three types of average dependency distances of sentences in a multilingual parallel corpus. In *Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation*, pages 652–661, 2021.

V. Padmakumar, B. Hedayatnia, D. Jin, P. L. Lange, S. Kim, N. Peng, Y. Liu, and D. Hakkani-Tur. Investigating the representation of open domain dialogue context for transformer models. In *Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 538–547, 2023.

O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

M. Pawelczyk, S. Neel, and H. Lakkaraju. In-context unlearning: Language models as few shot unlearners. *arXiv preprint arXiv:2310.07579*, 2023.

G. Penedo, Q. Malartic, D. Hesslow, R. Cojocaru, A. Cappelli, H. Alobeidli, B. Pannier, E. Almazrouei, and J. Launay. The refinedweb dataset for falcon llm: outperforming curated corpora with web data, and web data only. *ArXiv preprint*, abs/2306.01116, 2023. URL https://arxiv.org/abs/2306.01116.

G. Penedo, H. Kydlíček, L. B. allal, A. Lozhkov, M. Mitchell, C. Raffel, L. V. Werra, and T. Wolf. The fineweb datasets: Decanting the web for the finest text data at scale, 2024. URL https://arxiv.org/abs/2406.17557.

M. Pieler, M. Bellagente, H. Teufel, D. Phung, N. Cooper, J. Tow, P. Rocha, R. Adithyan, Z. Alyafeai, N. Pinnaparaju, et al. Rephrasing natural text data with different languages and quality levels for large language model pre-training. *arXiv preprint arXiv:2410.20796*, 2024.

E. Pizzi, S. D. Roy, S. N. Ravindra, P. Goyal, and M. Douze. A self-supervised descriptor for image copy detection. *Proc. CVPR*, 2022.

X. Qi, A. Panda, K. Lyu, X. Ma, S. Roy, A. Beirami, P. Mittal, and P. Henderson. Safety alignment should be made more than just a few tokens deep. *arXiv preprint arXiv:2406.05946*, 2024a.

X. Qi, A. Panda, K. Lyu, X. Ma, S. Roy, A. Beirami, P. Mittal, and P. Henderson. Safety alignment should be made more

than just a few tokens deep, 2024b. URL https://arxiv.org/abs/2406.05946.

X. Qiu, W. F. Shen, Y. Chen, N. Cancedda, P. Stenetorp, and N. D. Lane. PISTOL: Dataset compilation pipeline for structural unlearning of LLMs. *arXiv preprint arXiv:2406.16810*, 2024.

F. Radenovic, A. Dubey, A. Kadian, T. Mihaylov, S. Vandenhende, Y. Patel, Y. Wen, V. Ramanathan, and D. Mahajan. Filtering, distillation, and hard negatives for vision-language pre-training. *arXiv:2301.02280*, 2023.

A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models from Natural Language Supervision. In *International Conference on Machine Learning (ICML)*, 2021a.

A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021b.

R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36: 53728–53741, 2023.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020.

N. Rahman and E. Santacana. Beyond fair use: Legal risk evaluation for training llms on copyrighted text. 2023. URL https://genlaw.org/CameraReady/57.pdf.

A. Ramakrishna, Y. Wan, X. Jin, K.-W. Chang, Z. Bu, B. Vinzamuri, V. Cevher, M. Hong, and R. Gupta. LUME: LLM unlearning with multitask evaluations. *arXiv preprint arXiv:2502.15097*, 2025a.

A. Ramakrishna, Y. Wan, X. Jin, K.-W. Chang, Z. Bu, B. Vinzamuri, V. Cevher, M. Hong, and R. Gupta. SemEval-2025 Task 4: Unlearning sensitive content from large language models. *arXiv preprint arXiv:2504.02883*, 2025b.

B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning (ICML)*, 2019.

M. Roemmele, C. A. Bejan, and A. S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning.

In *AAAI Spring Symposium: Logical Formalizations of Commonsense Reasoning*, 2011.

R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

L. Rüschendorf. Random variables with maximum sums. *Advances in Applied Probability*, 14(3):623–632, 1982.

O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

O. Sainz, J. Campos, I. García-Ferrero, J. Etxaniz, O. L. de Lacalle, and E. Agirre. NLP evaluation in trouble: On the need to measure LLM data contamination for each benchmark. In H. Bouamor, J. Pino, and K. Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10776–10787, Singapore, 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.722. URL https://aclanthology.org/2023.findings-emnlp.722.

O. Sainz, J. A. Campos, I. García-Ferreroa, and J. E. andEneko Agirre. Did chatgpt cheat on your test? https://hitz-zentroa.github.io/lm-contamination/blog/, 2023b.

K. Sakaguchi, R. Le Bras, C. Bhagavatula, and Y. Choi. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of AAAI*, 2020.

K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106, 2021.

T. Sander, P. Fernandez, A. Durmus, M. Douze, and T. Furon. Watermarking makes language models radioactive. In A. Globersons, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. M. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/

2567c95fd41459a98a73ba893775d22a-Abstract-Conference.html.

T. Sander, P. Fernandez, S. Mahloujifar, A. Durmus, and C. Guo. Detecting benchmark contamination through watermarking. *ArXiv preprint*, abs/2502.17259, 2025. URL https://arxiv.org/abs/2502.17259.

M. Sap, H. Rashkin, D. Chen, R. LeBras, and Y. Choi. Social iqa: Commonsense reasoning about social interactions. In *Proceedings of EMNLP*, 2019.

A. Scherlis, K. Sachan, A. S. Jermyn, J. Benton, and B. Shlegeris. Polysemanticity and capacity in neural networks, 2025. URL https://arxiv.org/abs/2210.01892.

Y. Scholten, S. Günnemann, and L. Schwinn. A probabilistic perspective on unlearning and alignment for large language models. *arXiv preprint arXiv:2410.03523*, 2024.

C. Schuhmann, R. Vencu, R. Beaumont, R. Kaczmarczyk, C. Mullis, A. Katta, T. Coombes, J. Jitsev, and A. Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs, 2021.

C. Schuhmann, R. Beaumont, R. Vencu, C. W. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, P. Schramowski, S. R. Kundurthy, K. Crowson, and L. Schmidt. LAION-5B: An open large-scale dataset for training next generation image-text models. https://laion.ai/laion-5b-a-new-era-of-open-large-scale-multi-modal-datasets/, 2022.

A. Schwarzschild, Z. Feng, P. Maini, Z. Lipton, and J. Z. Kolter. Rethinking llm memorization through the lens of adversarial compression. *Advances in Neural Information Processing Systems*, 37:56244–56267, 2024a.

A. Schwarzschild, Z. Feng, P. Maini, Z. Lipton, and Z. Kolter. Rethinking LLM Memorization through the Lens of Adversarial Compression. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024b.

A. Schwarzschild, Z. Feng, P. Maini, Z. C. Lipton, and J. Z. Kolter. Rethinking llm memorization through the lens of adversarial compression. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 56244–56267. Curran Associates, Inc., 2024c. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/66453d578afae006252d2ea090e151c9-Paper-Confere pdf.

A. Sekhari, J. Acharya, G. Kamath, and A. T. Suresh. Remember what you want to forget: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34:18075–18086, 2021.

A. Seyitoğlu, A. Kuvshinov, L. Schwinn, and S. Günnemann. Extracting unlearned in-

formation from LLMs with activation steering. *arXiv preprint arXiv:2411.02631*, 2024.

H. Shah, K. Tamuly, A. Raghunathan, P. Jain, and P. Netrapalli. The pitfalls of simplicity bias in neural networks. *Advances in Neural Information Processing Systems*, 33: 9573–9585, 2020.

Z. Shen, T. Tao, L. Ma, W. Neiswanger, J. Hestness, N. Vassilieva, D. Soboleva, and E. Xing. Slimpajama-dc: Understanding data combinations for llm training. *arXiv preprint arXiv:2309.10818*, 2023.

Z. Shen, T. Tao, L. Ma, W. Neiswanger, Z. Liu, H. Wang, B. Tan, J. Hestness, N. Vassilieva, D. Soboleva, and E. Xing. Slimpajama-dc: Understanding data combinations for llm training, 2024. URL https://arxiv.org/abs/2309.10818.

T. Shi, K. Chen, and J. Zhao. Safer-instruct: Aligning language models with automated preference data. *arXiv preprint arXiv:2311.08685*, 2023a.

W. Shi, A. Ajith, M. Xia, Y. Huang, D. Liu, T. Blevins, D. Chen, and L. Zettlemoyer. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789*, 2023b.

W. Shi, A. Ajith, M. Xia, Y. Huang, D. Liu, T. Blevins, D. Chen, and L. Zettlemoyer. Detecting pretraining data from large language models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=zWqr3MQuNs.

W. Shi, A. Ajith, M. Xia, Y. Huang, D. Liu, T. Blevins, D. Chen, and L. Zettlemoyer. Detecting pretraining data from large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024b. URL https://openreview.net/forum?id=zWqr3MQuNs.

W. Shi, J. Lee, Y. Huang, S. Malladi, J. Zhao, A. Holtzman, D. Liu, L. Zettlemoyer, N. A. Smith, and C. Zhang. MUSE: Machine unlearning six-way evaluation for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TArmA033BU.

K. Shmelkov, C. Schmid, and K. Alahari. How good is my gan? In *Proceedings of the European conference on computer vision (ECCV)*, pages 213–229, 2018.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models, 2016. URL https://arxiv.org/abs/1610.05820.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017a.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, Los Alamitos, CA, USA, may 2017b. IEEE Computer Society. doi: 10.1109/SP.2017.41. URL https://doi.ieeecomputersociety.org/10.1109/SP.2017.41.

A. Shostack. The boy who survived: Removing Harry Potter from an LLM is harder than reported. *arXiv preprint arXiv:2403.12082*, 2024.

I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, and R. Anderson. The curse of recursion: Training on generated data makes models forget. 2024. URL https://arxiv.org/abs/2305.17493.

A. K. Singh, M. Y. Kocyigit, A. Poulton, D. Esiobu, M. Lomeli, G. Szilvasy, and D. Hupkes. Evaluation data contamination in llms: how do we measure it and (when) does it matter? *ArXiv preprint*, abs/2411.03923, 2024. URL https://arxiv.org/abs/2411.03923.

A. Sinitsin, V. Plokhotnyuk, D. Pyrkin, S. Popov, and A. Babenko. Editable neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJedXaEtvS.

L. N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE winter conference on applications of computer vision (WACV)*, pages 464–472. IEEE, 2017.

L. N. Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.

S. L. Smith, B. Dherin, D. Barrett, and S. De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=rq_Qr0c1Hyo.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, Oct. 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.

J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *ICML*, 2015. URL https://arxiv.org/abs/1503.03585.

L. Soldaini, R. Kinney, A. Bhagia, D. Schwenk, D. Atkinson, R. Authur, B. Bogin, K. Chandu, J. Dumas, Y. Elazar, V. Hofmann, A. H. Jha, S. Kumar, L. Lucy, X. Lyu, N. Lambert, I. Magnusson, J. Morrison, N. Muennighoff, A. Naik, C. Nam,

M. E. Peters, A. Ravichander, K. Richardson, Z. Shen, E. Strubell, N. Subramani, O. Tafjord, P. Walsh, L. Zettlemoyer, N. A. Smith, H. Hajishirzi, I. Beltagy, D. Groeneveld, J. Dodge, and K. Lo. Dolma: an Open Corpus of Three Trillion Tokens for Language Model Pretraining Research. *arXiv preprint*, 2024.

G. Somepalli, V. Singla, M. Goldblum, J. Geiping, and T. Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6048–6058, 2023a.

G. Somepalli, V. Singla, M. Goldblum, J. Geiping, and T. Goldstein. Understanding and mitigating copying in diffusion models. *Advances in Neural Information Processing Systems*, 36:47783–47803, 2023b.

G. Somepalli, A. Gupta, K. Gupta, S. Palta, M. Goldblum, J. Geiping, A. Shrivastava, and T. Goldstein. Measuring style similarity in diffusion models. *arXiv preprint arXiv:2404.01292*, 2024.

J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1): 2822–2878, 2018.

C. Stephenson, S. Padhy, A. Ganesh, Y. Hui, H. Tang, and S. Chung. On the geometry of generalization and memorization in deep neural networks, 2021. URL https://arxiv.org/abs/2105.14602.

Student. The probable error of a mean. *Biometrika*, pages 1–25, 1908.

D. Su, K. Kong, Y. Lin, J. Jennings, B. Norick, M. Kliegl, M. Patwary, M. Shoeybi, and B. Catanzaro. Nemotron-cc: Transforming common crawl into a refined long-horizon pretraining dataset. *arXiv preprint arXiv:2412.02595*, 2024.

Sustainable Technology Partners. Ai lawsuit timeline. https://sustainabletechpartner.com/topics/ai/generative-ai-lawsuit-timeline/, 2023. Accessed: 2023-09-25.

A. Talmor, J. Herzig, N. Lourie, and J. Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *Proceedings of NAACL*, pages 142–148, 2019.

L. Tang, P. Laban, and G. Durrett. MiniCheck: Efficient fact-checking of LLMs on grounding documents. In Y. Al-Onaizan,

M. Bansal, and Y.-N. Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8818–8847, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.499. URL https://aclanthology.org/2024.emnlp-main.499/.

R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

G. Team, R. Anil, S. Borgeaud, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, K. Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

N. Team, B. Adler, N. Agarwal, A. Aithal, D. H. Anh, P. Bhattacharya, A. Brundyn, J. Casper, B. Catanzaro, S. Clay, J. Cohen, et al. Nemotron-4 340B technical report. *arXiv preprint arXiv:2406.11704*, 2024.

P. Teterwak, C. Zhang, D. Krishnan, and M. C. Mozer. Understanding invariance via feedforward inversion of discriminatively trained classifiers. In *International Conference on Machine Learning*, pages 10225–10235. PMLR, 2021.

K. Thai, M. Karpinska, K. Krishna, B. Ray, M. Inghilleri, J. Wieting, and M. Iyyer. Exploring document-level literary machine translation with parallel paragraphs from world literature. In Y. Goldberg, Z. Kozareva, and Y. Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9882–9902, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.672. URL https://aclanthology.org/2022.emnlp-main.672.

P. Thaker, S. Hu, N. Kale, Y. Maurya, Z. S. Wu, and V. Smith. Position: LLM unlearning benchmarks are weak measures of progress. In *Proceedings of the 3rd IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, 2025.

A. Thudi, H. Jia, I. Shumailov, and N. Papernot. On the necessity of auditable algorithmic definitions for machine unlearning. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 4007–4022, 2022.

B. Tian, X. Liang, S. Cheng, Q. Liu, M. Wang, D. Sui, X. Chen, H. Chen, and N. Zhang. To forget or not? towards practical knowledge unlearning for large language models. In Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 1524–1537, Miami, Florida, USA, Nov. 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.82. URL https://aclanthology.org/2024.findings-emnlp.82/.

K. Tirumala, A. Markosyan, L. Zettlemoyer, and A. Aghajanyan. Memorization with-

out overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35: 38274–38290, 2022.

M. Toneva, A. Sordoni, R. T. d. Combes, A. Trischler, Y. Bengio, and G. J. Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

M. Toneva, A. Sordoni, R. T. des Combes, A. Trischler, Y. Bengio, and G. J. Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=BJlxm30cKm.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

E. Triantafillou, F. Pedregosa, J. Hayes, P. Kairouz, I. Guyon, M. Kurmanji, G. K. Dziugaite, P. Triantafillou, K. Zhao, L. S. Hosoya, J. C. S. J. Junior, V. Dumoulin, I. Mitliagkas, S. Escalera, J. Wan, S. Dane, M. Demkin, and W. Reade. Neurips 2023 - machine unlearning, 2023. URL https://kaggle.com/competitions/neurips-2023-machine-unlearning.

L. Tunstall, E. E. Beeching, N. Lambert, N. Rajani, K. Rasul, Y. Belkada, S. Huang, L. V. Werra, C. Fourrier, N. Habib, N. Sarrazin, O. Sanseviero, A. M. Rush, and T. Wolf. Zephyr: Direct distillation of LM alignment. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=aKkAwZB6JV.

E. Ullah, T. Mai, A. Rao, R. A. Rossi, and R. Arora. Machine unlearning via algorithmic stability. In *Conference on Learning Theory*, pages 4126–4142. PMLR, 2021.

E. Union. Regulation (eu) 2016/679 of the european parliament and of the council. *Official Journal of the European Union*, 2016.

United States Copyright Office. Copyright and artificial intelligence, part 3: Generative ai training. Technical report, United States Copyright Office, May 2025. URL https://www.copyright.gov/ai/. Pre-Publication Version.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

B. Vidgen, A. Agrawal, A. M. Ahmed, V. Akinwande, N. Al-Nuaimi, N. Alfaraj, E. Alhajjar, L. Aroyo, T. Bavalatti, M. Bartolo, et al. Introducing v0. 5 of the ai safety benchmark from mlcommons. *arXiv preprint arXiv:2404.12241*, 2024.

V. Vovk and R. Wang. Combining p-values via averaging. *Biometrika*, 107(4):791–808, 2020.

N. Vyas, S. M. Kakade, and B. Barak. On provable copyright protection for generative models. In *International Conference on Machine Learning*, pages 35277–35299. PMLR, 2023.

H. Wang, S. Ge, Z. Lipton, and E. P. Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

J. G. Wang, J. Wang, M. Li, and S. Neel. Pandora's white-box: Precise training data detection and extraction in large language models, 2024a. URL https://arxiv.org/abs/2402.17012.

P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, Y. Fan, K. Dang, M. Du, X. Ren, R. Men, D. Liu, C. Zhou, J. Zhou, and J. Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024b.

Q. Wang, B. Han, P. Yang, J. Zhu, T. Liu, and M. Sugiyama. Towards effective evaluations and comparisons for LLM unlearning methods. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=wUtCieKuQU.

Y. Wang, Q. Wang, F. Liu, W. Huang, Y. Du, X. Du, and B. Han. GRU: Mitigating the trade-off between unlearning and retention for large language models. *arXiv preprint arXiv:2503.09117*, 2025b.

Y. Wang, J. Wei, C. Y. Liu, J. Pang, Q. Liu, A. Shah, Y. Bao, Y. Liu, and W. Wei. LLM unlearning via loss adjustment with only forget data. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://openreview.net/forum?id=6ESRicalFE.

Z. Wang, C.-L. Li, V. Perot, L. Le, J. Miao, Z. Zhang, C.-Y. Lee, and T. Pfister. CodecLM: Aligning language models with tailored synthetic data. In K. Duh, H. Gomez, and S. Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3712–3729, Mexico City, Mexico, June 2024c. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-naacl.235. URL https://aclanthology.org/2024.findings-naacl.235/.

M. Weber, D. Y. Fu, Q. Anthony, Y. Oren, S. Adams, A. Alexandrov, X. Lyu, H. Nguyen, X. Yao, V. Adams, B. Athiwaratkun, R. Chalamala, K. Chen, M. Ryabinin, T. Dao, P. Liang, C. Ré, I. Rish, and C. Zhang. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*, 2024.

R. Webster, J. Rabin, L. Simon, and F. Jurie. Detecting gan-generated imagery using color cues. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4234–4238. IEEE, 2019.

A. Wei, N. Haghtalab, and J. Steinhardt. Jail-

broken: How does llm safety training fail? *arXiv preprint arXiv:2307.02483*, 2023.

J. T.-Z. Wei, R. Y. Wang, and R. Jia. Proving membership in llm pretraining data via data watermarks. *ArXiv preprint*, abs/2402.10892, 2024. URL https://arxiv.org/abs/2402.10892.

J. Welbl, N. F. Liu, and M. Gardner. Crowdsourcing multiple choice science questions. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, 2017.

Y. Wen, Y. Liu, C. Chen, and L. Lyu. Detecting, explaining, and mitigating memorization in diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

L. Weng. What are diffusion models? *lilianweng.github.io*, Jul 2021. URL https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

G. Wenzek, M.-A. Lachaux, A. Conneau, V. Chaudhary, F. Guzmán, A. Joulin, and E. Grave. Ccnet: Extracting high quality monolingual datasets from web crawl data. *arXiv preprint arXiv:1911.00359*, 2019.

A. Wettig, K. Lo, S. Min, H. Hajishirzi, D. Chen, and L. Soldaini. Organize the web: Constructing domains enhances pretraining data curation. *arXiv preprint arXiv:2502.10341*, 2025.

J. Wieting, K. Gimpel, G. Neubig, and T. Berg-kirkpatrick. Paraphrastic representations at scale. In W. Che and E. Shutova, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 379–388, Abu Dhabi, UAE, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-demos.38. URL https://aclanthology.org/2022.emnlp-demos.38.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

M. Wortsman, G. Ilharco, M. Li, J. W. Kim, H. Hajishirzi, A. Farhadi, H. Namkoong, and L. Schmidt. Robust fine-tuning of zero-shot models. *arXiv preprint arXiv:2109.01903*, 2021.

X. Wu, R. Duan, and J. Ni. Unveiling security, privacy, and ethical concerns of chatgpt. *Journal of Information and Intelligence*, 2023.

xAI. We will use grok 3.5 (maybe we should call it 4), which has advanced reasoning, to rewrite the entire corpus of hu-

man knowledge, adding missing information and deleting errors. June 2025. URL https://x.com/elonmusk/status/1936333964693885089.

S. Xhonneux, D. Dobre, M. Mofakhami, L. Schwinn, and G. Gidel. A generative approach to llm harmfulness detection with special red flag tokens, 2025. URL https://arxiv.org/abs/2502.16366.

H. Xia, Z. Yang, Q. Dong, P. Wang, Y. Li, T. Ge, T. Liu, W. Li, and Z. Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding, 2024.

H. Xu, N. Zhao, L. Yang, S. Zhao, S. Deng, M. Wang, B. Hooi, N. Oo, H. Chen, and N. Zhang. Relearn: Unlearning via learning for large language models. *arXiv preprint arXiv:2502.11190*, 2025.

Z. Xu, S. Yuan, L. Chen, and D. Yang. "a good pun is its own reword": Can large language models understand puns? In Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11766–11782, Miami, Florida, USA, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.657. URL https://aclanthology.org/2024.emnlp-main.657/.

Y. Xue, E. Moroshko, F. Chen, J. Sun, S. McDonagh, and S. A. Tsaftaris. Crce: Coreference-retention concept erasure in text-to-image diffusion models. *arXiv preprint arXiv:2503.14232*, 2025.

O. Yadan. Hydra - A framework for elegantly configuring complex applications. Github, 2019. URL https://github.com/facebookresearch/hydra.

B. Yang. CE-U: Cross Entropy unlearning. *arXiv preprint arXiv:2503.01224*, 2025.

Y. Yao, X. Xu, and Y. Liu. Large language model unlearning, 2023.

H. Ye, T. Liu, A. Zhang, W. Hua, and W. Jia. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794*, 2023.

S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282, Los Alamitos, CA, USA, jul 2018a. IEEE Computer Society. doi: 10.1109/CSF.2018.00027. URL https://doi.ieeecomputersociety.org/10.1109/CSF.2018.00027.

S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*, pages 268–282. IEEE, 2018b.

P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to vi-

sual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014. doi: 10.1162/tacl_a_00166. URL https://aclanthology.org/Q14-1006.

A. Zebaze, B. Sagot, and R. Bawden. Tree of problems: Improving structured problem solving with compositionality, 2024. URL https://arxiv.org/abs/2410.06634.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruyssen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, L. Beyer, O. Bachem, M. Tschannen, M. Michalski, O. Bousquet, S. Gelly, and N. Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark, 2020.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *The International Conference on Learning Representations*, 2017.

C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021a.

C. Zhang, D. Ippolito, K. Lee, M. Jagielski, F. Tramer, and N. Carlini. Counterfactual memorization in neural language models. In *NeurIPS*, 2021b. URL https://arxiv.org/abs/2112.12938.

C. Zhang, S. Bengio, and Y. Singer. Are all layers created equal? *Journal of Machine Learning Research*, 23(67):1–28, 2022. URL http://jmlr.org/papers/v23/20-069.html.

C. Zhang, D. Ippolito, K. Lee, M. Jagielski, F. Tramèr, and N. Carlini. Counterfactual memorization in neural language models. *Advances in Neural Information Processing Systems*, 36:39321–39362, 2023a.

H. Zhang, Y. Lin, and X. Wan. Pacost: Paired confidence significance testing for benchmark contamination detection in large language models, 2024a. URL https://arxiv.org/abs/2406.18326.

J. Zhang, J. Sun, E. Yeats, Y. Ouyang, M. Kuo, J. Zhang, H. F. Yang, and H. Li. Min-k%++: Improved baseline for detecting pre-training data from large language models. *ArXiv preprint*, abs/2404.02936, 2024b. URL https://arxiv.org/abs/2404.02936.

J. Zhang, J. Sun, E. Yeats, Y. Ouyang, M. Kuo, J. Zhang, H. F. Yang, and H. Li. Min-K%++:

Improved baseline for pre-training data detection from large language models. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=ZGkfoufDaU.

P. Zhang, G. Zeng, T. Wang, and W. Lu. Tinyllama: An open-source small language model, 2024c.

R. Zhang, L. Lin, Y. Bai, and S. Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *First Conference on Language Modelling*, 2024d. URL https://openreview.net/pdf?id=MXLBXjQkmb.

W. Zhang, M. Iwamura, M. Zeng, Y. Shi, and H. Kiya. Copyright protection in generative ai: A technical perspective. *arXiv preprint arXiv:2302.02333*, 2023b.

W. Zhang, R. Zhang, J. Guo, M. de Rijke, Y. Fan, and X. Cheng. Pretraining data detection for large language models: A divergence-based calibration method, 2024e. URL https://arxiv.org/abs/2409.14781.

X. Zhang, Y. Zhang, D. Long, W. Xie, Z. Dai, J. Tang, H. Lin, B. Yang, P. Xie, F. Huang, et al. mgte: Generalized long-context text representation and reranking models for multilingual text retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1393–1412, 2024f.

Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, et al. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023c.

Z. Zhang, F. Wang, X. Li, Z. Wu, X. Tang, H. Liu, Q. He, W. Yin, and S. Wang. Catastrophic failure of LLM unlearning via quantization. In *The Thirteenth International Conference on Learning Representations*, 2025b. URL https://openreview.net/forum?id=lHSeDYamnz.

C. Zhou, P. Liu, P. Xu, S. Iyer, J. Sun, Y. Mao, X. Ma, A. Efrat, P. Yu, L. YU, S. Zhang, G. Ghosh, M. Lewis, L. Zettlemoyer, and O. Levy. LIMA: Less is more for alignment. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=KBMOKmX2he.

C. Zhu, A. S. Rawat, M. Zaheer, S. Bhojanapalli, D. Li, F. Yu, and S. Kumar. Modifying memories in transformer models, 2021. URL https://openreview.net/forum?id=KubHAaKdSr7.

S. Zhu, R. Zhang, B. An, G. Wu, J. Barrow, Z. Wang, F. Huang, A. Nenkova, and T. Sun. Autodan: Automatic and interpretable adversarial attacks on large language models. *arXiv preprint arXiv:2310.15140*, 2023.

G. K. Zipf. *The psycho-biology of language: An introduction to dynamic philology*. Routledge, 2013.

A. Zou, Z. Wang, N. Carlini, M. Nasr, J. Z. Kolter, and M. Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A. Zou, L. Phan, J. Wang, D. Duenas, M. Lin, M. Andriushchenko, R. Wang, Z. Kolter, M. Fredrikson, and D. Hendrycks. Improving alignment and robustness with circuit breakers, 2024. URL https://arxiv.org/abs/2406.04313.