Memorization Without Overfitting: Analyzing the Training Dynamics of Large Language Models

Kushal Tirumala* **Aram H. Markosyan*** **Luke Zettlemoyer** **Armen Aghajanyan**

Meta AI Research
{ktirumala,amarkos,lsz,armenag}@fb.com

Abstract

Despite their wide adoption, the underlying training and memorization dynamics of very large language models is not well understood. We empirically study exact memorization in causal and masked language modeling, across model sizes and throughout the training process. We measure the effects of dataset size, learning rate, and model size on memorization, finding that larger language models memorize training data faster across all settings. Surprisingly, we show that larger models can memorize a larger portion of the data before over-fitting and tend to forget less throughout the training process. We also analyze the memorization dynamics of different parts of speech and find that models memorize nouns and numbers first; we hypothesize and provide empirical evidence that nouns and numbers act as a unique identifier for memorizing individual training examples. Together, these findings present another piece of the broader puzzle of trying to understand what actually improves as models get bigger.

1 Introduction

The rate and extent to which a model memorizes its training data are key statistics that provide evidence about how it is likely to generalize to new test instances. Classical frameworks, such as bias-variance tradeoff [1], argued for fitting a training set without full memorization. However, recent work has established a more symbiotic relationship between memorization and generalization in deep learning [2, 3, 4]. This paper empirically studies memorization in causal and masked language modeling, across model sizes and throughout the training process.

Memorization helps generalization

Much of the recent performance gains for language models have come from scale, with the most recent models reaching up to 10^{11} parameters [5, 6, 7]. Larger models are also known to memorize more training data [8], which is a crucial component of their improved generalization.

However, perhaps surprisingly, relatively little work has been done in understanding the impact of scale on the dynamics of language model memorization over training. Existing work focuses on analyzing memorization post-training [8, 9, 10, 11]. In this work, we study the memorization and forgetting dynamics in language models, with a focus on better measuring how they change as we scale up model size. Our primary contributions:

1. We characterize the dependence of memorization dynamics over training on model size (and other factors such as dataset size, overfitting, and learning rate). We find that larger language models memorize training data faster (§ 4).
2. We design controlled experiments that allow us to characterize the forgetting curves in language models (i.e., how language models naturally forget memories throughout training).

*Equal Contribution

Preprint. Under review.

Our empirical studies show that forgetting curves have lower bounds — we coin this as the *forgetting baseline* — and that this baseline increases with model scale, i.e., increasing model scale mitigates forgetting (§ 5).

3. We analyze the rates of memorization of different parts of speech, finding that nouns and numbers are memorized much more quickly than other parts of speech (§ 4.4). We hypothesize this is because the set of nouns and numbers can be seen as a unique identifier for a particular sample. We provide evidence to this hypothesis by analyzing the rates of memorization in the setting of an existing unique identifier (§ 4.3).

Together, these findings present another piece of the broader puzzle of trying to understand the unique training dynamics that emerge as models grow in size.

2 Background and Related Work

Memorization in Language Models: Unintended memorization is a known challenge for language models [12, 13], which makes them open to extraction attacks [14, 15] and membership inference attacks [16, 17], although there has been work on mitigating these vulnerabilities [11, 18]. Recent work has argued that memorization is not exclusively harmful, and can be crucial for certain types of generalization (e.g., on QA tasks) [19, 20, 21], while also allowing the models to encode significant amounts of world or factual knowledge [22, 23, 24]. There is also a growing body of work analyzing fundamental properties of memorization in language models [9, 8, 10]. Most related to our work [8] analyzes memorization of fully trained language models and observes a dependence on model scale, training data duplication, and prompting context length. While we also study scaling behavior, our focus instead is on the memorization dynamics throughout training.

Forgetting in Language Models: There has also been work studying memory degradation (forgetting) in language models. *Catastrophic forgetting* or *catastrophic interference*, first reported in [25, 26], studies how neural networks tend to forget the information from previous trained tasks or training batches, when trained on new data. This provides a key challenge for continual learning (or life-long learning) [27], where the goal is to gradually learn from a single pass over a, typically very large, stream of data. A number of mechanisms have been proposed for increasing robustness against catastrophic forgetting [28, 29, 30, 31, 32, 33]. There is also a growing body of work demonstrating that model scale and dataset scale can make models more resistant to forgetting [34, 35], as well as work characterizing how forgetting naturally occurs in image classifiers [36]. *Machine unlearning* is a technique that forces a trained model to forget a previously learned sample [37, 38], which is primarily motivated by data protection and privacy regulations [39, 40, 41, 42]. Our work is unique in its focus on measuring forgetting during training, and quantifying how it varies with scale.

Scaling Laws: We have consistently seen performance gains by scaling model size [5, 6, 7, 43, 44], and scale itself has been known to push internal model behavior away from classical bias-variance regimes [45]. Recent efforts have focused on trying to model the scaling laws for language models, including data and model size [46, 47], applications to transfer learning [48], routing networks [49], and various autoregressive generative tasks [50]. While the bulk of work in scaling laws has been empirical, an interesting line of work focuses on theoretically explaining neural scaling laws [51]. Most scaling laws focus exclusively on cross-entropy loss, while we instead study memorization as models scale, which we define formally in § 3.

3 Experimental Setup

In order to perform a large-scale study of the dynamics of memorization over training, our memorization metric must be reasonably easy to compute but also precise enough to tell us how much the model will actually remember from the training data. Label memorization is an ideal candidate, because it has consistently provided theoretical insight into underlying properties of neural networks, remains applicable in empirical settings, and is relatively cheap to compute. We formulate our metric as an analog of label memorization for self-supervised settings.

Definition 1 *Let V denote the vocabulary size. Let C denote a set of contexts, which can be thought of as a list of tuples (s, y) where s is an input context (incomplete block of text) and y is the index of the ground truth token in the vocabulary that completes the block of text. Let S denote the set of input* *contexts, and let $f : S \to \mathbb{R}^V$ denote a language model. A context $c = (s, y) \in C$ is memorized if* $\text{argmax}(f(s)) = y$.

Note that a single word can appear as the ground-truth token for multiple contexts. For a given set of contexts C (i.e a given training dataset), we can then analyze the proportion of memorized contexts

Proportion

$$\boxed{M(f) = \frac{\sum_{(s,y)\in C} \mathbb{1}\{\text{argmax}(f(s)) = y\}}{|C|}}$$

We refer to this as *exact memorization*, although it can also be seen as accuracy since we measure how often the argmax of the language model matches the ground truth token. Throughout this work, when we refer to memorization, we will be referring to Definition 1 unless we specify otherwise.

Number of times each datapoint has to be seen.

Will be a fraction ≤ 1

We define τ to be a threshold value for $M(f)$, and denote $T(N, \tau)$ as the minimal number of times a language model f with N parameter needs to see each training datapoint in order to satisfy $M(f) \geq \tau$. When leveraging bigger datasets, we introduce $T_{update}(N, \tau)$ as the minimal number of gradient descent updates U a language model f with N parameters needs to perform, to satisfy $M_{update}(f, U) \geq \tau$, where $M_{update}(f, U)$ is defined as the memorization on the batch of data on which the model performs the U'th gradient descent update.

Number of gradient updates needed

Previous work analyzing language modeling memorization defines memorization differently. Motivated by privacy concerns, both [8] and [14] define memorization from a training data extraction standpoint, in which a string s is extractable if it can be produced by interacting with the language model. More specifically, [14] defines a string s as being k-eidetic memorized if it is extractable and appears in at most k training examples. [8] defines a string s as k-memorized if the language model can produce it via prompting with k tokens of context from *training data*. This definition only works for causal language modeling because of the dependence on prompting with training data; for masked language modeling [8] uses Definition 1 above. Note that if an example is exactly memorized, it is extractable by definition. In other words, both the set of k-eidetic memorized tokens and the set of k-memorized tokens contain the set of exactly memorized tokens (formally, different exactly memorized tokens may be contained in different sets, depending on k). Therefore, analyzing exact memorization gives a type of lower bound on the k-eidetic memorization and k-memorization. In a different line of work motivated by estimating the influence of individual training examples, [9] defines a training example x as memorized if the difference in expected model performance (where model performance is defined as $M(f)$ above) over subsets of data including x and subsets of data not including x, is sufficiently large. This definition pulls from previous work in theoretically analyzing label memorization in classification settings [52].

Different definitions of memorization in different works.

Model Architectures: We replicate publicly available references for Transformer language model architectures [53, 54]. We use the 125 million, 355 million, 1.3 billion, 2.7 billion, 6.7 billion, and 13 billion model configurations (see § A.4 for more explicit architecture and hyperparameter configurations). We fix these architectures across all experiments. We train using the FairSeq framework [55] with PyTorch [56] as the underlying framework. For our larger models, we use the fully sharded data-parallel implementation available in FairScale [57] and use Aim experiment tracking [58].

Didn't know about these

Datasets: We use two existing datasets across all our experiments: the WIKITEXT-103 benchmark containing around 103 million tokens [59], and the RoBERTa corpus [60] used to train the original RoBERTa model, containing around 39 billion tokens (we refer to this as the ROBERTA dataset).

4 Larger Language Models Memorize Faster

Larger neural language models are known to be more sample efficient and require fewer optimization steps to reach the same performance [46] while also converging faster [61], where performance is usually defined as *test* perplexity. In this section, we study $T(N, \tau)$ on the *training* set as a function of N to answer this question.

In the left plot of Figure 1, we fix a memorization threshold $\tau = 0.9$ and examine $T(N, \tau)$ as we increase N. We see that larger language models need to see each training datapoint fewer times to achieve 90% exact memorization of the training set; in other words, $T(N, 0.9)$ is monotonically decreasing in N. When we vary τ between 0.4 and 0.95 in the right plot of Figure 1, we still observe

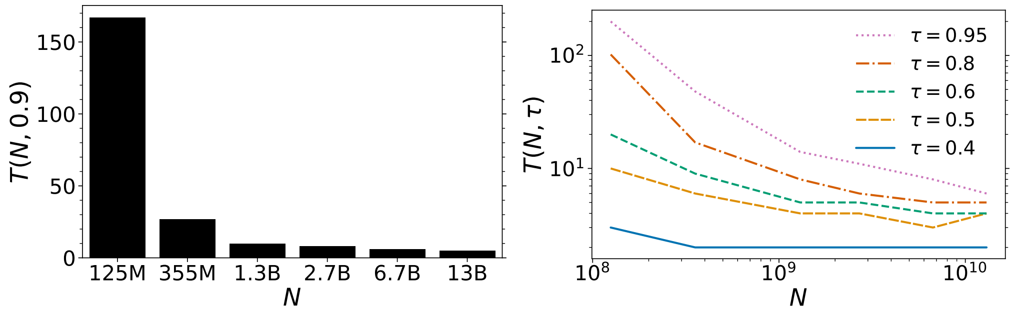
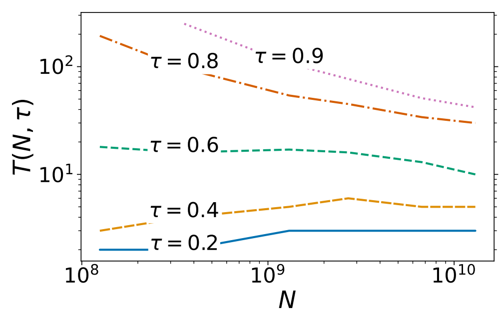


Figure 1: We show $T(N, \tau)$, which is the number of times a language model needs to see each training example before memorizing τ fraction of the training data, as a function of model size N. Result are for causal language modeling on WIKITEXT103, right plot is on log-log scale. Note that generally larger models memorize faster, regardless of τ.

N is number of parameters in model

that $T(N, \tau)$ is generally monotonically decreasing with N.[2] We note that for fixed N, $T(N, \tau)$ is increasing in τ, which is expected since memorizing more of the training set requires training the model for more epochs. More interestingly, increasing τ smoothly transitions $T(N, \tau)$ from constant in N, to exponentially decreasing in N (the axes are on a log-log scale).



For masked LM

Figure 2: $T(N, \tau)$ as a function of N (shown on log-log scale), for various values of τ in masked language modeling on WIKITEXT103. We show that larger models initially memorize training data slower, but reach high proportions of training data memorization faster.

4.1 Dependence on Language Modeling Task and Dataset Size

To investigate the dependence of our observations on the particular language modeling task, we repeat this analysis for the masked language modeling task on WIKITEXT103 with mask probability 0.15. Unlike in causal language modeling, Figure 2 shows that $T(N, \tau)$ is not monotonically decreasing in N for lower values of τ, and is monotonically decreasing in N for higher values of τ, where the phase transition between these two regimes occurs between $\tau = 0.6$ and $\tau = 0.7$. Smaller models memorize the training data quicker initially and slower in the long run (e.g., right plot of Figure 11).

Language model training is heavily dependent on the dataset size [46], and therefore we expect $M(f)$ to be similarly impacted. In Figure 3, we analyze training set memorization on the much bigger ROBERTA dataset for both masked and causal language modeling. With large datasets such as ROBERTA dataset, it becomes infeasible to perform multiple epochs and evaluate memorization on the entire training set, especially when training larger models. Consequently, we focus on smaller values of τ and investigate the number of gradient descent updates it takes to reach memorization thresholds, i.e., $T_{update}(N, \tau)$. In Figure 3 we observe a similar trend as Figure 1, where $T_{update}(N, \tau)$ is monotonically decreasing with N for various τ, in both masked and causal language modeling.

[2] We fix 0.4 as the lower bound for the range because any lower value for the memorization threshold is achieved within the first few epochs across all model scales (the line in Figure 1 is essentially flat), and 0.95 as the upper bound because higher values require unreasonably long training time for smaller models.

=> No. of updates required decreases with N for various τ for both causal & masked LM.

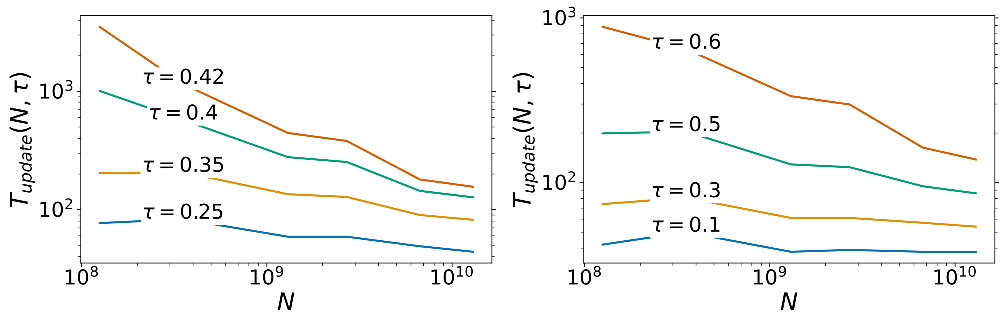


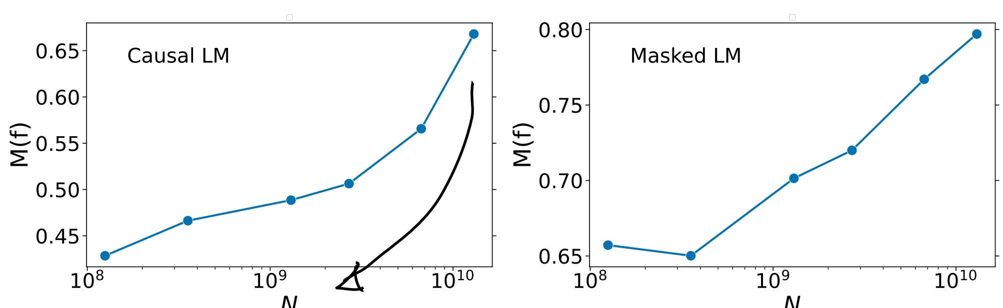
Figure 3: We show $T_{update}(N, \tau)$, which is the number of gradient descent updates U a language model needs to perform before memorizing τ fraction of the data given on the U'th update, as a function of model size N. Result are for causal (Left) and masked (Right) language modeling on the ROBERTA dataset, on a log-log scale. We show that larger models memorize faster, regardless of τ.

Unlike with WIKITEXT103, it seems that masked language modeling does not have a phase transition for τ.

4.2 Why Do Larger Models Memorize Faster?

A natural question at this point is to ask why larger models memorize faster? Typically, memorization is associated with overfitting, which offers a potentially simple explanation. In order to disentangle memorization from overfitting, we examine memorization before overfitting occurs, where we define overfitting occurring as the first epoch when the perplexity of the language model on a validation set increases. Surprisingly, we see in Figure 4 that as we increase the number of parameters, memorization before overfitting generally increases, indicating that overfitting by itself *cannot* completely explain the properties of memorization dynamics as model scale increases.

LM



For N = 10^{10}, 65% memorization still occurs before

Figure 4: Proportion of training data memorized $M(f)$ before overfitting, as a function of model size N (plotted on a log scale). Results are for causal (left) and masked (right) language modeling on WIKITEXT103. Note that larger models memorize more before overfitting.

The learning rate is not constant across our training configurations. Intuitively, larger learning rates should lead to quicker memorization. To investigate to what extent our results can be explained by learning rate, we take a subset of the architectures available above and train on the WIKITEXT103 dataset across a standard range of learning rates while measuring memorization, in Figure 5. Even if we fix a learning rate, larger models reach 0.9 memorization faster, suggesting that our results are not caused solely by differences in learning rates. Interestingly, sensitivity to learning rate generally decreases as we increase the model size. We also notice in Figure 5 that $T(N, \tau)$ goes down initially (for low LRs) and eventually rises (for high LRs), and as the long as the chosen learning rate places us in this "basin", the memorization dynamics do not change significantly (note that axes are on log-scale). This result is consistent with the growing intuition that for neural language models past a particular scale, the learning rate is not a significant hyperparameter [54, 62]. I have seen this as well experimentally.

Exhaustively searching all such possible factors is intractable, and providing a complete explanation for why larger models memorize faster is outside the scope of this work. However, it is possible that

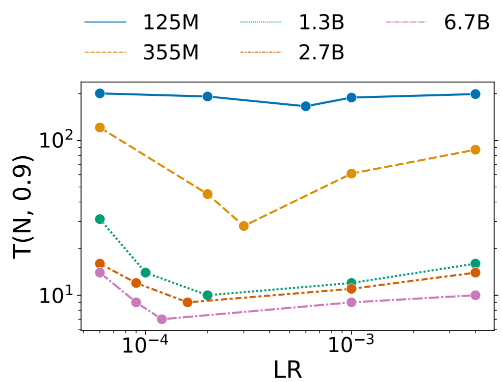


Figure 5: Examining the effect of learning rate (LR) on number of times model needs to see each training example in order to reach 0.9 proportion of training data memorization $T(N, 0.9)$. Each line corresponds to a different model size performing causal language modeling on WIKITEXT103. We demonstrate that larger models memorize faster for a fixed learning rate.

why is there such a trend??

classical ML concepts cannot even explain such a memorization trend. The following two sections introduce novel studies that we hope will expand the toolkit available for research on analyzing memorization.

4.3 Memorization via. Unique Identifiers

Recent work studies how to use external memory to improve performance [19, 20, 21, 24]. In this subsection, we question whether such architecture changes are necessary. Motivated by information retrieval systems, we take a simple approach — we prepend a unique identifier to every example in the training set and examine whether memorization speed increases. Specifically, we fix the language modeling task as causal language modeling on WIKITEXT103 with the 125M parameter model, and in front of every training example, we insert the string document ID <unique_id> where unique_id is a unique integer, one for each training context. In order to utilize all these unique integers, we must add them to the dictionary of tokens, which causes a significant increase in the model size since the last layer in the language model must have an output dimension equal to the size of the dictionary. To control for this, we examine how the dynamics of $M(f)$ change when simply increasing the dictionary size by adding unique identifiers (without ever using any of the added tokens) and then examine the added effect on $M(f)$ when utilizing those unique identifiers. In Figure 6, we see that increasing the dictionary size does improve the speed of memorization. Even though we previously demonstrated that larger models memorize faster, this is still surprising considering that we do not increase parameter size in a significant way — we are effectively adding fake tokens to the dictionary. Moreover, when we leverage those added tokens to identify training examples uniquely, we see yet another gain in memorization, although prompting using a document ID shifts memorization dynamics away from being monotonically increasing over time.

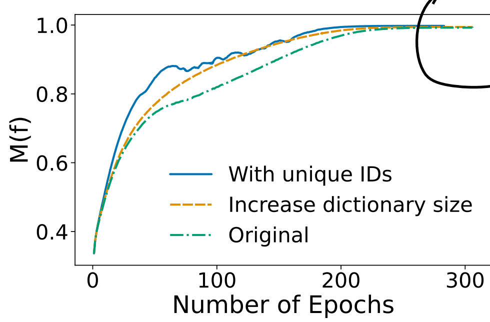
Adding unique identifiers while Prompting increase speed of memorization?

4.4 Memorization Through the Lens of Parts of Speech

In the previous section, we showed that a unique identifier enhances memorization. Regular text also contains strong proxies to unique identifiers in the form of numerals and proper nouns. Motivated by this, we study syntactic features of memories using part-of-speech (POS) tagging.[3] We track the ratio $R(p)$ of the number of positions for which the part of speech p was correctly predicted to the total number of tokens in the ground truth tagged with that part-of-speech p (see the left plot in Figure 7). In other words, $R(p)$ can be thought of as tracking *memorization of parts of speech*. In the right plot of Figure 7 we show a similar ratio, denoted $R_{mem}(p)$, but the numerator only considers the tokens that are also exactly memorized. The correctly predicted part of speech does not necessarily imply exact memorization, which is clearly illustrated by Figure 7 where we see the language model memorizing parts of speech faster than the exact value of the token. While all parts of speech are

POS tag easier to memorize than exact token.

[3] We use spaCy [63] to identify parts of speech in a text.



However it seems to converge towards the end.

Figure 6: The impact of adding unique identifiers to training examples on memorization $M(f)$ training dynamics for causal language modeling on WIKITEXT103 at the 125M model scale. The green line is the original 125M model. The orange line is the model after adding unique identifiers to the dictionary (which increases model size). The blue line prepends these unique identifiers for each training example. Note that adding unique identifiers leads to faster memorization of training data.

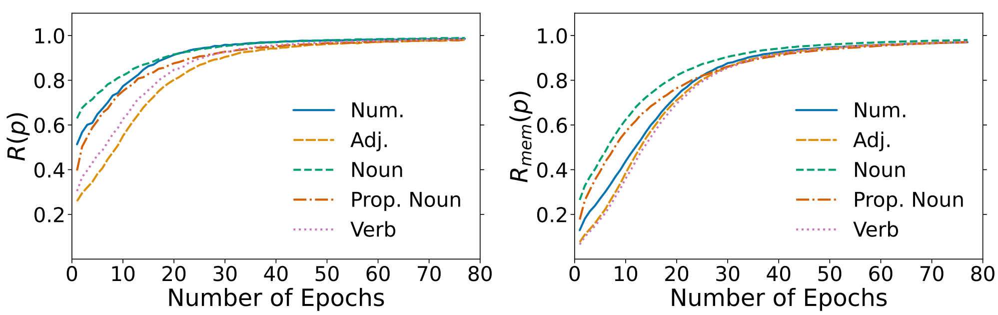


Figure 7: The ratios $R(p)$ (Left) and $R_{mem}(p)$ (Right) over training. $R(p)$ represents proportion of POS correctly memorized (the language model outputs the right POS, but not necessarily the correct word). $R_{mem}(p)$ represents the proportion of exactly memorized tokens for a particular POS p. Results are for causal language modeling on WIKITEXT103 at the 355M model scale. In both plots, we consider numerals, proper nouns, verbs, nouns, and adjectives as potential parts of speech (i.e., values for p). We show that nouns and numerals are memorized faster than other parts of speech.

learned, *some parts of speech are memorized faster*. Specifically, nouns, proper nouns, and numerals are memorized noticeably faster than verbs and adjectives, both in terms of $R(p)$ and $R_{mem}(p)$. This has implications for privacy since sensitive information is likely to be a noun/proper noun/numeral. Our findings also roughly align with work studying child language acquisition [64].

See more work along this (if any) ☆

5 Forgetting Curves in Language Models

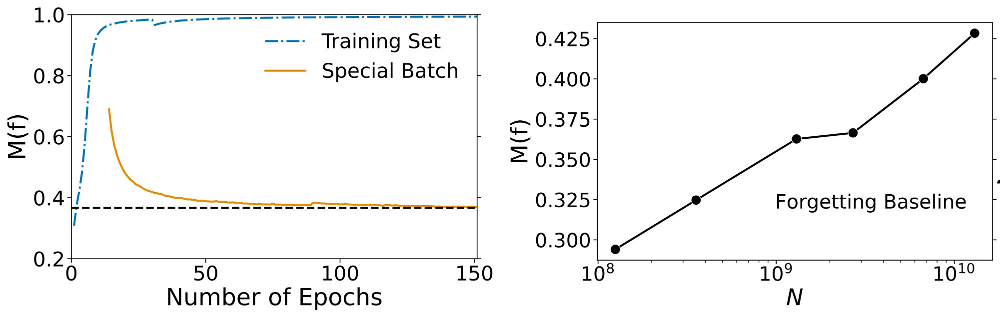
This section studies the dual side of memorization — forgetting in language models. Inspired by the *forgetting curve* hypothesis, according to which human memory declines over time when there is no attempt to retain it [65], we are interested in understanding the dynamics of memory degradation in language models.

We first choose a batch of data not available in the training set, i.e., a batch of data from a validation set. We refer to this batch of data as the *special batch*. We then take a checkpoint from model training, plug in the special batch so that the model can train on it, and resume standard training on the training set. We then evaluate how memorization degrades on the special batch and analyze the various factors the forgetting curve may depend on. We use the entire validation set as the special batch throughout this section. This experimental setup is different from catastrophic forgetting, as we fix the data distribution by pulling the special batch from the same dataset as the training set. Similarly, this setup differs from machine unlearning since we are not explicitly devising an algorithm to remove information from a language model; instead, we analyze how forgetting naturally occurs. It is also

Train on something (special batch) for a bit and then see the forgetting curve.

Similar concepts that are actually different
— Catastrophic Forgetting
— Machine Unlearning
— Intrinsic Hallucination

different from intrinsic hallucination [66]; wherein there is an assumption that contradicted output is semantically correct (e.g., the language model outputs a wrong date)



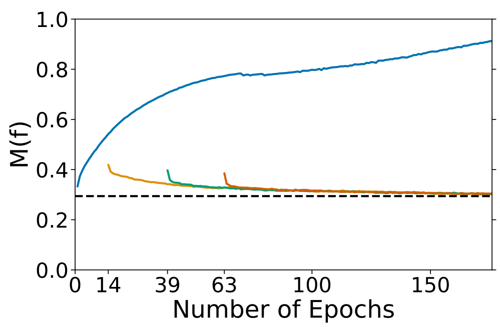
Large models forget less

Figure 8: Left: forgetting curve for 2.7B model size for causal language modeling on WIKITEXT103. The dashed horizontal line indicates the lowest proportion of special batch data memorized throughout training, i.e., the forgetting baseline. Right: forgetting baseline as a function of model size N (plotted on log scale). We show that as model scale increases, the forgetting baseline value increases.

In the left plot of Figure 8, we show the forgetting curve for the 2.7B model. Exact memorization on the special batch degrades quickly at first, but slows down exponentially as we continue training [4] (see Figure 15 in § A.2.2). In other words, the forgetting curve on the special batch seems to approach a baseline — we refer to this trend as the *forgetting baseline*. We approximate the forgetting baseline by looking at the lowest memorization value on the special batch throughout training.

We show the forgetting baseline as a function of the model scale in the right plot of Figure 8. We see that the numerical value for the baseline is monotonically increasing with the model scale. This implies that larger models forget less, aligning with recent work studying catastrophic forgetting on image classification tasks [35]. This is beneficial because larger models can leverage more information from previous tasks; however, from a privacy perspective, this is not ideal because it implies larger models may be potentially retaining more sensitive information from training data.

and after seeing it not that many times



Always reaches the forgetting baseline in the end.

Figure 9: We empirically show that the forgetting baseline does not depend on data batch ordering. We inject the special batch into the training set at the 14th, 39th, and 63rd epochs, and evaluate proportion of special batch data memorized as we continue training. Results are for causal language modeling on WIKITEXT103 at the 125M model scale.

We also investigate the sensitivity of the forgetting baseline on data batch order. In Figure 9, we perform the same forgetting curve analysis described above but start the analysis at different training checkpoints (in Figure 9 we start at the 14th, 39th, and 63rd epochs). This way, we alter the order of the data batches given to the model (since the special batch will appear in a different place in the global order of data batches given to the model) without drastically changing the experimental setup. We observe that the forgetting baseline is not sensitive to data batch order.[5]

[4]The average sequential difference in memorization (on the special batch) on the last 3 epochs of training is at most on the order of 10^{-3}, whereas the average sequential difference in the first 3 epochs of training is consistently on the order of 10^{-2}

[5]The max difference between the numerical values for the baseline are on the order of 10^{-3}

Motivated by *replay methods* from continual learning (see [30] for a survey) and work in promoting retention memories in humans such as *repetition* and *spaced repetition* [67, 68, 69], we study the effect of repetition and spaced repetition on the forgetting baseline in the left and right plots of Figure 10. In the left plot, we inject the special batch into the training set multiple times before continuing training on the training set alone. We observe that the forgetting baseline is monotonically increasing as a function of repetition frequency (differences in the baseline value are on the order of 10^{-2}). To study the spaced repetition, we periodically inject the held-out set into the training set, train on it once, and then continue training on the training set alone. We notice in the right plot of Figure 10 that spaced repetition incurs minimal effect on the forgetting baseline (on the order of 10^{-3}), independent of the length of spacing between the repetitions.

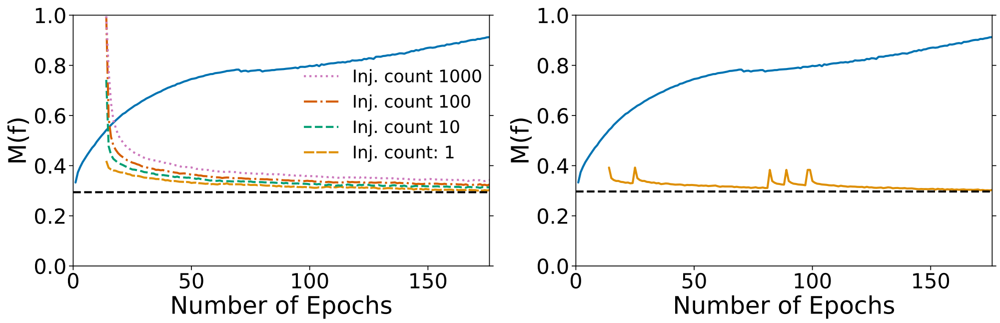


Figure 10: Effect of repeated injection of special batch (Left) and spaced repetition (Right) for various spacing intervals on special batch memorization. Results are for causal language modeling (125M) on WIKITEXT103. The solid upper curve represents the training set memorization. We show that repeated injection increases the forgetting baseline, whereas spaced repetition has minimal effect.

An exciting direction for future work will be to understand the structure of the baseline — for example, understanding what types of tokens (parts of speech, synonyms, facts, syntax) are memorized in the baseline and the overlap of tokens memorized in the baseline with tokens in the training set.

6 Conclusions and Discussion

We study the properties of memorization dynamics over language model training and demonstrate that larger models memorize faster. We also measure the properties of forgetting curves and surprisingly find that forgetting reaches a baseline, which again increases with the model scale. Combined with memorization analyses that expose the unintuitive behavior of language models, we hope to motivate considering memorization as a critical metric when increasing language model scale.

Most work studying memorization in language modeling is primarily motivated by privacy (see § 2). While theoretically, there are well-established frameworks to quantify privacy such as differential privacy [70], empirical privacy in language modeling is not well-defined — does memorizing common knowledge count as information leakage? Does outputting a synonym count as harmful memorization? As per our Definition 1, we implicitly focus on information that is sensitive if outputted verbatim (phone numbers, SSNs, addresses, medical diagnoses, etc.), rather than capturing all aspects of privacy. It is also known that text data used for training language models contain certain biases and stereotypes (e.g., [71]); therefore, our work has similar implications for how long language models can train before they definitively memorize these biases from training data.

We also hope our work highlights the importance of analyzing memorization dynamics as we scale up language models, instead of only reporting cross entropy. Cross-entropy loss and memorization capture different behavior — for example, in many of our memory degradation experiments, even though memorization approaches a baseline, we observe that perplexity is still increasing (see Figure 14 in § A.2 for an example). This implies that the model is becoming unconfident about the exact predictions, which we can only conclude because we inspect loss and memorization. More importantly, the forgetting baseline behavior would be entirely obscured if we did not inspect memorization dynamics. Similarly, there are multiple instances where we uncover interesting behavior *because* we focus on memorization dynamics (§ 4.4, § 4.3, § A.3), rather than focusing only on cross-entropy loss.

Lot more interesting results in the Appendix!

7 Acknowledgements

The authors would like to thank Chuan Guo, Alex Sablayrolles, Pierre Stock, and Adina Williams for helpful discussions throughout the course of this project. The authors would also like to researchers at FAIR who commented on or otherwise supported this project, including Shashank Shekhar, Candace Ross, Rebecca Qian, Dieuwke Hupkes, and Gargi Ghosh.

References

[1] James Franklin. The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, 27(2):83–85, 2005.

[2] Vitaly Feldman. Does learning require memorization. *A short tale about a long tail. CoRR, abs/1906.05271*, 2019.

[3] Vitaly Feldman and Chiyuan Zhang. What neural networks memorize and why: Discovering the long tail via influence estimation. *Advances in Neural Information Processing Systems*, 33:2881–2891, 2020.

[4] Gavin Brown, Mark Bun, Vitaly Feldman, Adam Smith, and Kunal Talwar. When is memorization of irrelevant training data necessary for high-accuracy learning? In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 123–132, 2021.

[5] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[6] Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

[7] Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*, 2022.

[8] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

[9] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual Memorization in Neural Language Models. *arXiv:2112.12938 [cs]*, December 2021. arXiv: 2112.12938 version: 1.

[10] Eugene Kharitonov, Marco Baroni, and Dieuwke Hupkes. How bpe affects memorization in transformers. *arXiv preprint arXiv:2110.02782*, 2021.

[11] Om Thakkar, Swaroop Ramaswamy, Rajiv Mathews, and Françoise Beaufays. Understanding unintended memorization in federated learning. *arXiv preprint arXiv:2006.07490*, 2020.

[12] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[13] Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 196–206, 2019.

[14] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[15] Aleena Thomas, David Ifeoluwa Adelani, Ali Davody, Aditya Mogadala, and Dietrich Klakow. Investigating the impact of pre-trained word embeddings on memorization in neural networks. In *International Conference on Text, Speech, and Dialogue*, pages 273–281. Springer, 2020.

[16] Sorami Hisamoto, Matt Post, and Kevin Duh. Membership inference attacks on sequence-to-sequence models. *arXiv preprint arXiv:1904.05506*, 2019.

[17] Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. *arXiv preprint arXiv:2203.03929*, 2022.

[18] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[19] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*, 2019.

[20] Yi Tay, Vinh Q Tran, Mostafa Dehghani, Jianmo Ni, Dara Bahri, Harsh Mehta, Zhen Qin, Kai Hui, Zhe Zhao, Jai Gupta, et al. Transformer memory as a differentiable search index. *arXiv preprint arXiv:2202.06991*, 2022.

[21] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2021.

[22] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[23] Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*, 2022.

[24] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020.

[25] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[26] Roger Ratcliff. Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*, pages 285–308, 1990.

[27] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

[28] Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. *arXiv preprint arXiv:2004.12651*, 2020.

[29] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[30] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[31] Wojciech Masarczyk, Kamil Deja, and Tomasz Trzcinski. On robustness of generative representations against catastrophic forgetting. In *International Conference on Neural Information Processing*, pages 325–333. Springer, 2021.

[32] Chenze Shao and Yang Feng. Overcoming catastrophic forgetting beyond continual learning: Balanced training for neural machine translation. *arXiv preprint arXiv:2203.03910*, 2022.

[33] Armen Aghajanyan, Akshat Shrivastava, Anchit Gupta, Naman Goyal, Luke Zettlemoyer, and Sonal Gupta. Better fine-tuning by reducing representational collapse. *arXiv preprint arXiv:2008.03156*, 2020.

[34] Seyed Iman Mirzadeh, Arslan Chaudhry, Huiyi Hu, Razvan Pascanu, Dilan Gorur, and Mehrdad Farajtabar. Wide neural networks forget less catastrophically. *arXiv preprint arXiv:2110.11526*, 2021.

[35] Vinay Venkatesh Ramasesh, Aitor Lewkowycz, and Ethan Dyer. Effect of scale on catastrophic forgetting in neural networks. In *International Conference on Learning Representations*, 2021.

[36] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

[37] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

[38] Yang Liu, Zhuo Ma, Ximeng Liu, Jian Liu, Zhongyuan Jiang, Jianfeng Ma, Philip Yu, and Kui Ren. Learn to forget: Machine unlearning via neuron masking. *arXiv preprint arXiv:2003.10933*, 2020.

[39] Elizabeth Liz Harding, Jarno J Vanto, Reece Clark, L Hannah Ji, and Sara C Ainsworth. Understanding the scope and impact of the california consumer privacy act of 2018. *Journal of Data Protection & Privacy*, 2(3):234–253, 2019.

[40] Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

[41] Alessandro Mantelero. The eu proposal for a general data protection regulation and the roots of the 'right to be forgotten'. *Computer Law & Security Review*, 29(3):229–235, 2013.

[42] General Data Protection Regulation. General data protection regulation (gdpr). *Intersoft Consulting, Accessed in October*, 24(1), 2018.

[43] Armen Aghajanyan, Bernie Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, et al. Cm3: A causal masked multimodal model of the internet. *arXiv preprint arXiv:2201.07520*, 2022.

[44] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[45] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

[46] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models. *arXiv:2001.08361 [cs, stat]*, January 2020. arXiv: 2001.08361.

[47] Jonathan S Rosenfeld, Amir Rosenfeld, Yonatan Belinkov, and Nir Shavit. A constructive prediction of the generalization error across scales. *arXiv preprint arXiv:1909.12673*, 2019.

[48] Danny Hernandez, Jared Kaplan, Tom Henighan, and Sam McCandlish. Scaling laws for transfer. *arXiv preprint arXiv:2102.01293*, 2021.

[49] Aidan Clark, Diego de las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. *arXiv preprint arXiv:2202.01169*, 2022.

[50] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. Scaling laws for autoregressive generative modeling. *arXiv preprint arXiv:2010.14701*, 2020.

[51] Yasaman Bahri, Ethan Dyer, Jared Kaplan, Jaehoon Lee, and Utkarsh Sharma. Explaining neural scaling laws. *arXiv preprint arXiv:2102.06701*, 2021.

[52] Vitaly Feldman. Does Learning Require Memorization? A Short Tale about a Long Tail. *arXiv:1906.05271 [cs, stat]*, January 2021. arXiv: 1906.05271.

[53] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

[54] Mikel Artetxe, Shruti Bhosale, Naman Goyal, Todor Mihaylov, Myle Ott, Sam Shleifer, Xi Victoria Lin, Jingfei Du, Srinivasan Iyer, Ramakanth Pasunuru, et al. Efficient large scale language modeling with mixtures of experts. *arXiv preprint arXiv:2112.10684*, 2021.

[55] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*, 2019.

[56] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[57] Mandeep Baines, Shruti Bhosale, Vittorio Caggiano, Naman Goyal, Siddharth Goyal, Myle Ott, Benjamin Lefaudeux, Vitaliy Liptchinsky, Mike Rabbat, Sam Sheiffer, Anjali Sridhar, and Min Xu. Fairscale: A general purpose modular pytorch library for high performance and large scale training. `https://github.com/facebookresearch/fairscale`, 2021.

[58] Gor Arakelyan, Gevorg Soghomonyan, and The Aim team. Aim, 6 2020.

[59] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *ArXiv*, abs/1609.07843, 2017.

[60] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv e-prints*, July 2019.

[61] Zhuohan Li, Eric Wallace, Sheng Shen, Kevin Lin, Kurt Keutzer, Dan Klein, and Joey Gonzalez. Train big, then compress: Rethinking model size for efficient training and inference of transformers. In *International Conference on Machine Learning*, pages 5958–5968. PMLR, 2020.

[62] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[63] Matthew Honnibal and Ines Montani. spaCy 3: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2022.

[64] Michael Fleischman and Deb Roy. Why verbs are harder to learn than nouns: Initial insights from a computational model of intention recognition in situated word learning. In *27th Annual Meeting of the Cognitive Science Society, Stresa, Italy*, 2005.

[65] Geoffrey R Loftus. Evaluating forgetting curves. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 11(2):397, 1985.

[66] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *arXiv preprint arXiv:2202.03629*, 2022.

[67] Paul Smolen, Yili Zhang, and John H Byrne. The right time to learn: mechanisms and optimization of spaced learning. *Nature Reviews Neuroscience*, 17(2):77–88, 2016.

[68] Shiri Oren, Charlene Willerton, and Jeff Small. Effects of spaced retrieval training on semantic memory in alzheimer's disease: A systematic review. *Journal of Speech, Language and Hearing Research (Online)*, 57(1):247, 2014.

[69] Jeffrey D Karpicke and Henry L Roediger III. Expanding retrieval practice promotes short-term retention, but equally spaced retrieval enhances long-term retention. *Journal of experimental psychology: learning, memory, and cognition*, 33(4):704, 2007.

[70] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[71] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Realtoxicityprompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020.

[72] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499*, 2021.

[73] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

[74] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[75] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[76] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017.

A Appendix

A.1 Full Memorization Dynamics Over Training

For completeness, in this section we plot our memorization metric $M(f)$ over training for all model sizes. In any of these plots, observe that taking a horizontal slice for a fixed τ is equivalent to computing $T(N, \tau)$. In Figure 11, we plot $M(f)$ over training for WIKITEXT103. We see that generally (across language modeling tasks and and values of τ), larger models memorize faster. We do notice a caveat in Figure 11, where we observe that in initial stages of training, smaller models memorize faster, but larger models eventually surpass smaller models.

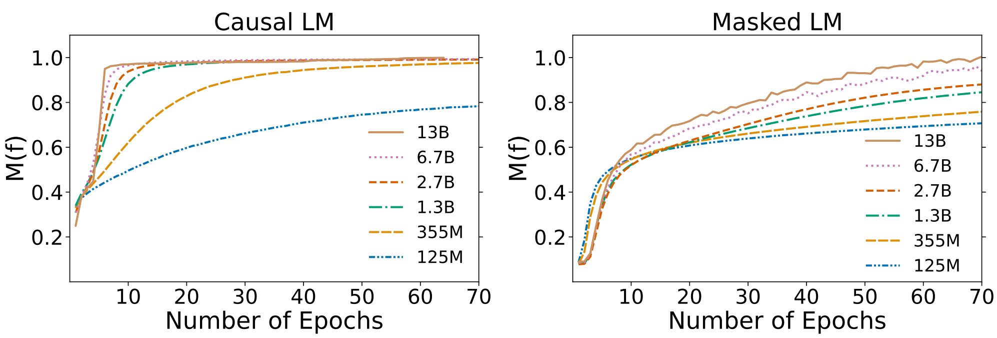


Figure 11: Proportion of training data memorized $M(f)$ over training, for causal (Left) and masked (Right) language modeling on WIKITEXT103. The x-axis describes the number of epochs, and y-axis denotes $M(f)$ as defined in § 3. Generally, we see that larger models memorize training data faster.

When we analyze larger datasets, performing multiple epochs of training becomes infeasible, and so we track memorization with each gradient descent update. Similarly, we cannot analyze $M(f)$ for the entire training dataset. We use notation introduce in § 1, specifically $M_{update}(f, U)$ where U is the number of gradient updates performed on model f. This quantity is defined as the memorization on the batch of data given to the model on the U'th update. In figure 12, we take a rolling average with window size 5 when plotting $M_{update}(f, U)$ to smooth out curves.

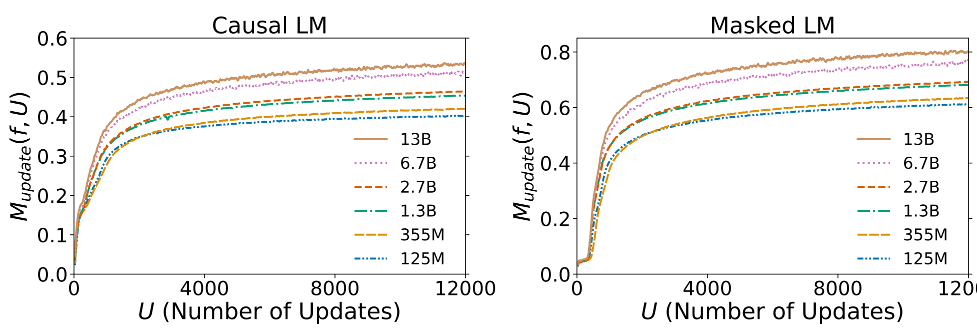


Figure 12: Proportion of training data memorized $M(f)$ over training, for causal (Left) and masked (Right) language modeling on the ROBERTA dataset. The x-axis describes the number of gradient descent updates, and the y-axis denotes a rolling average (window size 5) of $M_{update}(f)$ as defined above. We again notice that larger models memorize training data faster.

To check that $M_{update}(f, U)$ is a viable proxy for $M(f)$, in Figure 13, we plot both $M(f)$ and $M_{update}(f, U)$ up to 30000 updates for two model sizes. We fix 30000 as the upper bound, because we only train some model sizes up to 30000 updates in the ROBERTA experiments in § 4, and therefore can only completely assess the impact of scale on $M_{update}(f, U)$ dynamics up to 30000 updates. We see that $M_{update}(f, U)$ has periodic behavior, but overall does not deviate too much from $M(f)$.

When it becomes expensive $M_{update}(f, U)$ is a good proxy for $M(f)$

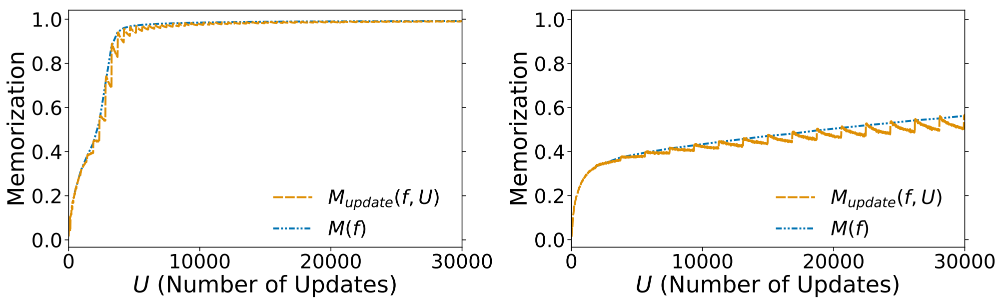


Figure 13: We show training data memorization evaluated at the end of an epoch $M(f)$, and at the end of each gradient descent update $M_{update}(f, U)$, over training. Results shown are for causal language modeling on WikiText103 dataset for 13B (Left) and 125M (Right) model sizes. We note that $M_{update}(f, U)$ closely tracks $M(f)$ throughout training.

A.1.1 Limitations of Definition 1

We note that Definition 1 is not the best way to study memorization: it ignores model confidence and it does not normalize for duplication in the training set (it is known that duplication in the training set helps models memorize tokens [8, 72]). However, as mentioned in Section 3, all previous definitions of memorization seem to involve Definition 1 in some form. In this way, we study a metric fundamental to memorization regardless of the precise definition of memorization.

A.2 Forgetting Baseline Analysis

A.2.1 Perplexity Versus Memorization

This section shows how perplexity and memorization on the special batch evolve over training. In Figure 14 we see that perplexity continues to increase over training, while memorization flatlines. This is a clear experimental setup where we find cross-entropy loss capturing different behavior from memorization. We show plots for the 1.3B model scale, although all of the experiments in § 5 exhibit very similar trends.

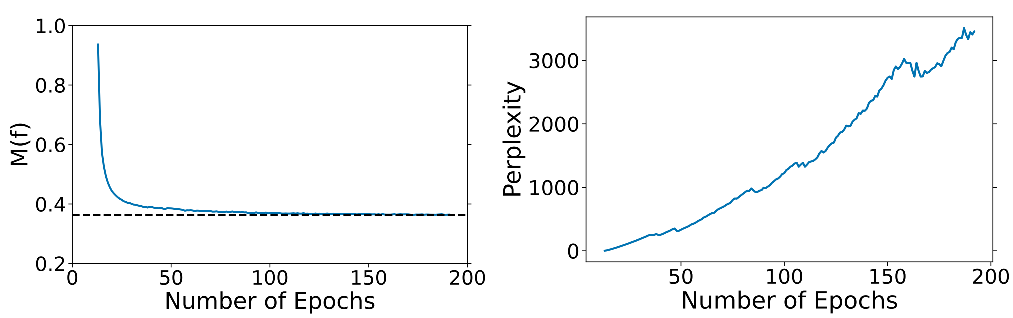


Figure 14: Proportion of special batch data memorized $M(f)$ (Left) and perplexity of special batch (Right) in the forgetting baseline experimental setup described in § 5. Results are for causal language modeling on WikiText103 with 1.3B model size. We notice that memorization of the special batch flattens, while perplexity continues increasing.

A.2.2 Verifying Existence of Baseline

To verify the existence of the forgetting baseline discussed in § 5, we observe the sequential difference in $M(f)$ of the special batch, from epoch to epoch. More formally, if $M(f)_T$ denotes the memorization at epoch T, we investigate $\texttt{diff}(T) = M(f)_T - M(f)_{T-1}$ on the special batch, for $T > 1$. In Figure 15 we show this plot for a few model scales, and we clearly see that the sequential difference in $M(f)$ exponentially approaches 0.

Sequential diff reaches 0, implying a baseline.

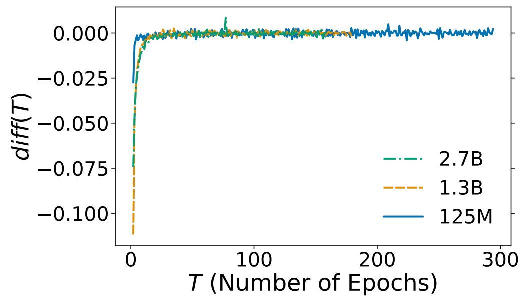


Figure 15: Exploring the sequential difference in proportion of training data memorized $M(f)$ on the special batch over training. The x-axis denotes the number of epochs (i.e. T) and the y-axis denotes the sequential difference in $M(f)$ from the $(T-1)$'th epoch to the T'th epoch (i.e $\texttt{diff}(T)$). Results shown are for causal language modeling on WIKITEXT103. We show that sequential difference in memorization exponentially approaches 0.

A.3 Analyzing Memory Unit Length Over Training

This section investigates a fundamental property of memories — memory unit length L. We look at individual tokens memorized as having length $L = 1$, memorized bigrams as having length $L = 2$, memorized trigrams as having length $L = 3$, etc. Analyzing memory length is interesting because it has implications for how language models retain n-grams, which are an important part of language. Moreover, recent work shows that chain-of-thought prompting improves language model performance [73]; understanding memory unit length informs us whether a similar method might work for improving performance when training (if a language model has low memory unit length, then including chain-of-thought-type texts in the training set might not have a significant effect). An empirical side note is that these experiments were run separately from the main paper experiments, so we provide original $M(f)$ curves for reference.

Chain of thought prompting can help when memory unit length is high.

We track the average value of L across the entire training dataset for causal language modeling on WIKITEXT103. Note that in our all our experiments, the sequence length is constrained to be less than 512 tokens, with an average sequence length of 430.12 on WIKITEXT103. In the left plot of Figure 16 we analyze the average memory unit length over training for two model sizes. We observe across model sizes that average memory unit length steadily increases over time, roughly taking a sigmoidal shape. We notice that the larger 2.7B model has an average L increasing faster than the 125M model. This is consistent with our previous results because we know larger models memorize, and some of these tokens are likely to be adjacent to each other, especially as the model achieves higher values of $M(f)$. Surprisingly, we see that the average memory unit length is much lower than the average sequence length of 430.12, suggesting that even with high individual token memorization (which is achieved as shown in the right plot of Figure 16), there are always tokens in the middle of a text that the language model has not yet memorized, which break up the memories.

The average unit length memorized is still less than the average seq. length, implying that some tokens must be "forgotten" in the middle.

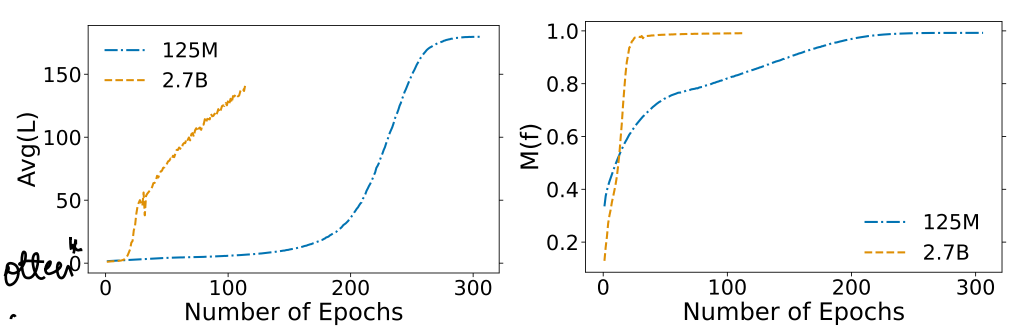


Figure 16: Left: Examining average memory unit length L (averaged over the entire training dataset), as function of number of epochs. As a reference, we show the memorization dynamics $M(f)$ on the right. Results shown are for causal language modeling on WIKITEXT103.

A.4 Model Training/Dataset Details

In this section, we layout the details of experiments, although most training details we pull directly from publicly available references [53, 54]. As such, we provide the details of model architectures using the same style as Table 1 in [53] for ease of comparison. All models use GELU activation [74] for nonlinearity. We leverage the Adam optimizer [75], with $\beta_1 = 0.9$, $\beta_2 = 0.98$, and $\epsilon = 10^{-8}$. For reproducibility, we set weight decay to 0, dropout to 0, and attention dropout to 0. We use a polynomial learning rate schedule, and following [53, 54] we scale up our learning rate from 0 to the maximum learning rate over $375M$ tokens, and scale down to 0 over the remaining $T - 375M$ tokens (for all masked language modeling experiments, and all ROBERTA experiments, we have $T = 300B$; for causal language modeling experiments on WIKITEXT103 we have $T = 100B$). We fix a sequence length of 512 across all experiments, but we break input text up into complete sentences, so not all input texts have length exactly equal to 512. In masked language modeling experiments, we use a mask probability of 0.15. When training language models, we use the standard procedure of minimizing cross-entropy loss, and use dynamic loss scaling [76].

→ Mixed precision training

Table 1: Model architecture details. # L denotes the number of layers, # H denotes the number of attention heads, and d_{model} denotes embedding size. Global batch size denotes the total number of tokens the model processes in a batch of data. Note that most of the values in this table are the same as Table 1 in [53].

Model Scale	# L	# H	d_{model}	Learning Rate (LR)	Global Batch Size
125M	12	12	768	6.0e-4	0.5M
355M	24	16	1024	3.0e-4	0.5M
1.3B	24	32	2048	2.0e-4	1M
2.7B	32	32	2560	1.6e-4	1M
6.7B	32	32	4096	1.2e-4	2M
13B	40	40	5120	1.0e-4	2M

As mentioned in § 3, we use FairSeq [55] which relies on PyTorch [56]. When training models, we leverage fully sharded data-parallel implementation of models in FairScale [57]. We utilize NVIDIA A100 GPUs with 40GB of memory. Increasing model scale requires different amounts of GPUS: 125M and 355M generally required 16 GPUS, 1.3B required 32 GPUS, and 2.7B, 6.7B, and 13B generally required 64 GPUS (although some experiment runs were launched with 128 GPUS in order to decrease training time). Exact training time varied depended on model scale and dataset size, but all models were trained for up to 140 hours.

In both datasets we use, there is a possibility for sensitive or offensive text to be included in the training set, since both benchmarks use data that is scraped from the Internet. We also note that the WIKITEXT103 benchmark we use throughout the work is available under the Creative Commons Attribution-ShareAlike License. The ROBERTA dataset we use refers to the corpora of text originally used to train the RoBERTa model (see [60]). This dataset not publicly available under any license, however subsets of data that make up the corpus are publicly available.