Uncovering Latent Memories: Assessing Data Leakage and Memorization Patterns in Large Language Models

Sunny Duan [1] **Mikail Khona** [1] **Abhiram Iyer** [1] **Rylan Schaeffer** [2] **Ila Fiete** [1]

Abstract

The proliferation of large language models has revolutionized natural language processing tasks, yet it raises profound concerns regarding data privacy and security. Language models are trained on extensive corpora including potentially sensitive or proprietary information, and the risk of data leakage — where the model response reveals pieces of such information — remains inadequately understood. This study examines susceptibility to data leakage by quantifying the phenomenon of memorization in machine learning models, focusing on the evolution of memorization patterns over training. We reproduce findings that the probability of memorizing a sequence scales logarithmically with the number of times it is present in the data. Furthermore, we find that sequences which are not apparently memorized after the first encounter can be "uncovered" throughout the course of training even without subsequent encounters. The presence of these "latent" memorized sequences presents a challenge for data privacy since they may be hidden at the final checkpoint of the model. To this end, we develop a diagnostic test for uncovering these latent memorized sequences by considering their cross entropy loss.

1. Introduction

Large language models (LLMs) are trained on vast data-sets (Touvron et al., 2023; Gemini Team et al., 2023; OpenAI et al., 2023; Brown et al., 2020). The size of the training datasets enables high competency in the trained models in the sense of fluency, knowledge about various domains (AlKhamissi et al., 2022; Guu et al., 2020), and the ability to perform in-context reasoning. The training datasets often include proprietary, copyrighted, or otherwise private information. In human memory, repeated encounters with information and data are gradually transformed from an "episodic" or contextually detailed verbatim-like stores into "semantic" stores in which the gist and general nature of the content is retained but the specifics are discarded. (Tulving, 1972)

In contrast, LLMs are capable of not only using training data for general knowledge and performance, but have been shown to possess a vast capacity for detailed memorization. Specifically, with appropriate cue-ing, LLMs can reproduce verbatim text from their training corpii. This phenomenon is the opposite of "catastrophic forgetting", in which shifts in the training data cause models to forget previous learning, which has led to a vigorous subfield of research on mitigating this interference-driven forgetting (Kirkpatrick et al., 2017; Zenke et al., 2017; De Lange et al., 2019; Serrà et al., 2018; Wang et al., 2023; Schwarz et al., 2018; Ritter et al., 2018). In part, the ability of LLMs to exhibit detailed memory of training data may be due to their large size. Yet LLMs are often trained on a single pass through the data corpus, meaning that the model encounters distributional shifts throughout training. Surprisingly, the verbatim recall of LLMs extends to sequences seen early in training (Biderman et al., 2023b).

1.1. Related Work

Extracting memorized sequences from language models is an area of high interest. Early work established that it was possible to extract sensitive data including phone numbers, URLs and personal information from trained language models (Zanella-Béguelin et al., 2019). Other studies injected canaries to determine which aspects of the training process contributed to whether a sequence can be extracted (Henderson et al., 2017; Thakkar et al., 2020). More recent work has extended this to investigate how these properties scale with model size and data statistics (Carlini et al., 2022a; Karamolegkou et al., 2023; Ozdayi et al., 2023; Carlini et al., 2022b).

The definition of memorization is also still debated and various approaches to quantifying memorization have been

[1]Massachusetts Institute of Technology, Massachusetts, USA [2]Department of Computer Science, Stanford University, California, USA. Correspondence to: Sunny Duan <sunnyd@mit.edu>.

Proceedings of the 41^{st} *International Conference on Machine Learning*, Vienna, Austria. PMLR 235, 2024.

made (Zhang et al., 2021; Feldman & Zhang, 2020). A variety of attacks have been designed to extract memorized sequences using designed prompts (Thakkar et al., 2020) and model activation perturbations (Kassem et al., 2024).

Finally, there has also been work studying how the training process affects the status of memorization (Tirumala et al., 2022). This work focuses on how parameters of training and size of the model affect the dynamics of training. They find that scaling the model generally leads to less forgetting. In our work, we focus on sequences which counter-intuitively do not obey the forgetting laws presented in this work and expanding on the implications of these persistent ”episodic” memories.

1.2. Contribution

This work provides insights into the dynamics and mechanics of memorization in large language models, contributing to the broader understanding of data privacy and security within machine learning. Our primary contributions are as follows:

Quantification of Memorization Susceptibility: We systematically evaluate how the statistical characteristics of training data, specifically sequence complexity and repetition, influence the likelihood of memorization in language models.

Stationarity of Memorized Sequences: We discover that the memorization status of sequences remains largely stationary after initial exposure, despite not being reencountered.

Latent Memorization and Recovery: We identify the presence of ”latent” memorized sequences, which are not evident at certain checkpoints but can be uncovered later in training or through controlled perturbations.

Development of a Diagnostic Test: We propose a novel diagnostic test for uncovering latent memorized sequences by analyzing their cross-entropy loss. This test provides a practical tool for detecting and mitigating potential data leakage in deployed language models.

2. Methodology

2.1. Sequence Complexity

As in previous studies (Carlini et al., 2020), we find that one class of data which is highly represented in memorized data are ”simple” sequences such as repeated subsequences, sequences of numbers or highly structured sequences.

This notion of complexity can be formalized using Kologomorov complexity. Kologomorov complexity is defined as the minimum description needed to reproduce a sequence. While this formalism is helpful, it is a theoretical measure which cannot be computed readily. As a proxy, we use modern compression algorithms to determine the extent to which sequences can be reproduced from a smaller description. In order to calculate the complexity of a sequence we define a metric z-compressibility which is the ratio between the compressed length of the sequence and the length of the original sequence. This metric is an upper bound on the Kologomorov complexity of the sequence since the Kologomorov complexity is defined as the smallest of such descriptions.

2.2. Quantifying memorization

One popular definition of memorization is kl-memorization (Carlini et al., 2022a). kl-memorization is evaluated by considering a sequence of length $k + l$. The first k tokens are presented to the model as context. The model is used to generate a continuation of length l. The model's continuation is compared to the ”true” continuation, and a sequence is said to be kl memorized if the model's output exactly matches the true continuation.

We find that kl-memorization may be overly strict. In many cases, the model may make small errors such as inserting or modifying a single token which results in a ”forgotten” sequence 1. In order to mitigate this, we propose a modification of kl-memorization by introducing k-Levenshtein distance (k-LD) in which k context tokens are provided to the model and the measure of memorization is given by the Levenshtein distance (edit distance) between the true continuation and the model continuation. We find that this is a more natural measure of memorization which also provides a range of values which produces more granular insight into the strength of the model's memory. Throughout this study, we set $k = 32$ and compare the continuation of the model with the original sequence by computing the levenshtein distance between the next 64 tokens.

2.3. Analyzing repeated Samples

We analyze where training sequences were repeated throughout the course of training. In our study, we focus on the l portion of the sequence. For this study, we fixed l to be 64 tokens. Given this target sequence, we compare the target sequence with all of the training sequences which were presented to the model during the period of training under consideration. We compute the largest subsequence match between 512,000 target sequences and every individual training example and call a training example a ”repeat” if there was a sub-sequence match of length 30 or longer.

2.4. Models

In this study, we used the model, Pythia-1b (Biderman et al., 2023a), which was trained on the Pile dataset(Gao et al., 2020). For selected experiments, we reproduced the results using a larger and better performing model, Amber-7B (Liu et al., 2023), in order to ensure that our results were consistent with other large language models. We selected these two models as they were large high performing models which had fully reproducible data sequences and frequent checkpoints. As in previous works (Biderman et al., 2023a), all experiments were run with the models run with half precision and no temperature.

3. Experimental results

3.1. Statistics of memorization

We analyze two primary drivers of memorization during training: sequence complexity, and the number of repetitions. Previous studies have shown that the probability of extraction is correlated with number of repetitions (Carlini et al., 2020). We are able to reproduce this result in our data as well (Figure 1a). In addition, we found that the z-complexity of the string itself was a strong predictor of whether a sequence was memorized (Figure 1b). We found that for strings of different complexity exhibited different memorization curves (Figure 1c). Both of these factors influenced the memorization probability with a log-linear relationship.

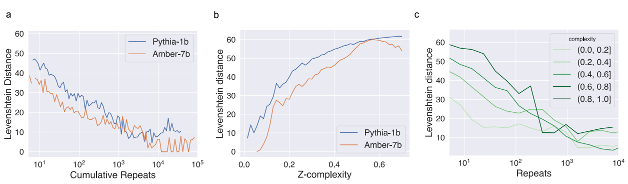


Figure 1. **Data statistics and the probability of memorization a.** Plot of average k-LD as a function of the number of times the sequence is repeated in the dataset for Pythia-1b and Amber-7b **b.** Average k-LD as a function of the Z-complexity of the sequence. **c.** Relationship between k-LD and repeats for different complexity levels. **d.** Comparison of the predictions of the best linear model predicting the k-LD from the logarithm of the sequence complexity and number of repetitions.

3.2. Dynamics of memorization

We analyze how the k-LD changes throughout the course of training for individual sequences. In order to eliminate the effects of repeated exposure, we filter out sequences which are repeated according to our heuristic outlined above. For these target sequences, we track the k-LD for the sequence and measure how it changes as training proceeds. Surprisingly, we find that the memorization status of a sequence is largely stationary throughout training. After the initial checkpoint, the k-LD of the sequences fluctuate but do so in a way which is stable across training (Figure 2d). This

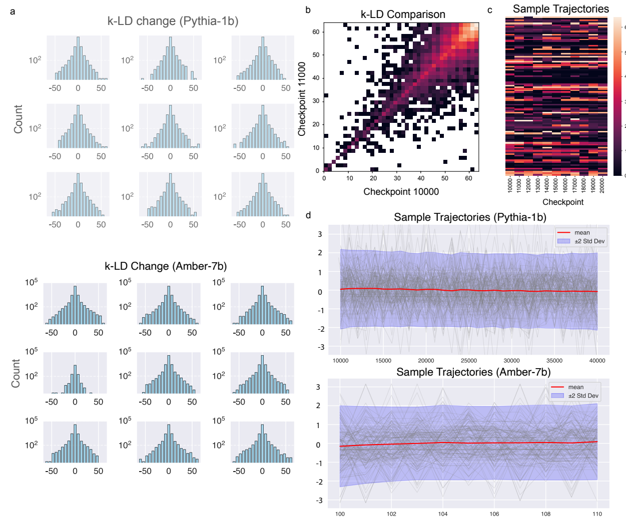


Figure 2. **Memorization status is stationary a.** Histograms of changes of edit distance between consecutive checkpoints for sequences which were encountered once during training. **b.** Distribution of k-LD during checkpoint 10k and 11k. Color is the log of the number of sequences in each bin. The vast majority of sequences are not memorized in either checkpoint. **c.** Visualization of individual samples and the change in the memorized length during training. **d.** Grey lines are sub-sampled single sequence trajectories throughout training. Each trajectory of k-LD measurements was normalized such that the distribution of k-LD was mean 0 and variance 1 over the period of interest. The red line denotes the mean of all sequences under consideration and shaded area denotes region of two standard deviations of k-LD at a given poitn in time..

is consistent in both Pythia-1b and Amber-7b models. The memorization status for individual sequences as well as the population mean show no clear trend as training progresses. Furthermore, unlike a random walk, we see that the variance of the does not grow over time, but remains fixed. This is indicative of a mean reversion tendency of the dynamics and demonstrate the stability of the memories within the model weights. Additionally, we observe that the changes in the k-LD between consecutive checkpoints (Figure 2ab) are symmetric and roughly follow a laplace distribution. This again confirms the counter-intuitive property of sequences to become memorized as often as they are forgotten. Notably, the model is able to recall memories which, at one point in time, appeared to be forgotten, despite never encountering that sequence again.

The stationarity of the memorization status of these sequences indicates that the memorized sequence is fixed throughout time, but this is in conflict with the fact that the model weights are constantly evolving. This stability in the presence of noise is indicative of a stabilizing mechanism by which the encoding of the sequence memory is preserved by a restorative process illustrated where the memorized sequence becomes a fixed point in the weight space of the model under training dynamics. Subsequent training may alter the readout of the sequence, but the memory of the sequence is fixed throughout time. Since this is not true of all sequences, it may point to a phase transition that occurs when the sequence is first encountered.

3.3. Latent memorization and recovery

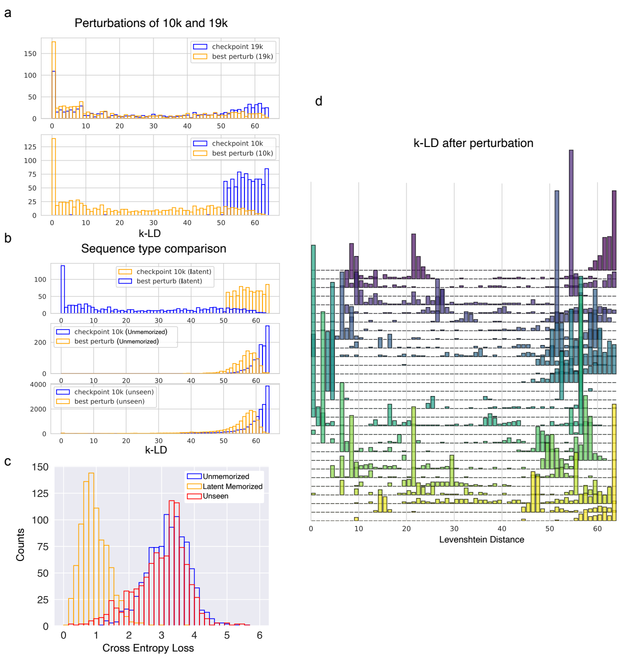


Figure 3. **a.** Comparison of the distribution of best achievable k-LD by perturbing the model weights. Top panel is the histogram of the perturbations of the model at checkpoint 19k and bottom is the model at checkpoint 10k. **b.** Comparison of using perturbations to evoke a target sequence for three different classes of sequences. In the top panel, we examine the sequences which are "latent" memorized. In the middle panel, we find sequences which weren't memorized during training and in the bottom panel, we analyze sequences which were encountered later in training but were not encountered by the model. We not that perturbing the weights is only able to evoke sequences which are "latent" memorized. **c.** Comparison of the cross entropy losses of sequences separated into the three different classes of sequences analyzed in b. The cross entropy losses of "latent" memorized sequences are much lower. **d.** Drawing of a mechanistic proposal for how memorization is stabilized during training. **e.** Visualization of the Levenshtein distances from the target for various perturbations. Each row is a single sequence, and the heights of the bars correspond to the number of perturbations which resulted in a Levenshtein distance of the corresponding bin.

We analyzed how the memorization status of sequences which were encountered between checkpoint 9k and 10k, changed later on in training. We found many sequences which were not memorized at 10k (k-LD > 50) but subsequently became memorized later on in training and exhibited memorization by checkpoint 19k Table 1 despite never being encountered again. For these sequences, the nature of the random changes shown in figure 2 indicate the form of a random walk. We hypothesize that the process of training in large language models acts as random noise on the weights with respect to the memory of the sequence. Thus, simply perturbing the weights with random noise should produce similar effects as training does.

We test this by randomly perturb the model weights by adding a small amount of random isotropic gaussian noise with $\sigma = 10^{-3}$ to each of the weight parameters. This noise was roughly an order smaller than the weights themselves. We repeat this process 200 times and select perturbation which yields the lowest k-LD.

We find that sequences which were not memorized at checkpoint 10k but were memorized later in training were able to be recovered by random perturbation (Figure 3a). As a control, we also considered sequences which were not presented to the model by checkpoint 20k, and observed that the distribution of k-LD closely matched those which were encountered by not memorized by the model (Figure 3b). Furthermore, the distribution of minimum k-LD for the perturbations of the model at 10k closely resembled the distribution of the k-LD for the model at checkpoint 19k. These observations indicate subsequent training acts similar to random noise perturbations to the model weights.

These sequences which are not memorized at one point in training but appear later seem to be "remembered" by the model in spite of their incorrect continuation. They can be considered to be "latent" memorized as they may not be visible at the current point in training, but they can be uncovered by small perturbations of the weights. These sequences pose a significant risk for leakage since they are not easily detectable from evaluating kl-memorization. To this end, we discovered that these "latent" memorized sequences had significantly lower cross entropy loss when evaluated by the model (Figure 3c), thus simply evaluating the likelihood of those sequences using the trained model is a natural diagnostic for detecting these "latent" memorized sequences.

4. Conclusion and limitations

We study how memorization changes throughout training and focused on sequences which occurred only once throughout training. Under these conditions, we find that rather than forgetting these sequences, the model retains them for the duration of training. We characterized the nature of memorization changes throughout training using random weight perturbations. These perturbations confirm that sequences which appeared to be forgotten at one point during training, may still be memorized by the model and are able to be uncovered with a small amount of random noise. We concluded by demonstrating a simple diagnostic to distinguish between "latent" memorized sequences and un-memorized sequences.

This study highlights one surprising behavior of large language models and begins to elucidate what mechanisms are present in the memorization behavior of these models. Our work suggest a possible mechanism of how memorized strings are sustained throughout training and further experiments are needed to confirm the underlying mechanism. Notably, further testing is required across other large language models which were not considered here. We also propose a mechanistic explanation for this phenomenon which requires additional experiments to explain the cause of these persistent memories.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

AlKhamissi, B., Li, M., Celikyilmaz, A., Diab, M. and Ghazvininejad, M. A review on language models as knowledge bases. April 2022.

Biderman, S., Prashanth, U.S., Sutawika, L., Schoelkopf, H., Anthony, Q., Purohit, S. and Raff, E. Emergent and predictable memorization in large language models. April 2023a.

Biderman, S., Prashanth, U.S., Sutawika, L., Schoelkopf, H., Anthony, Q.G., Purohit, S. and Raf, E. Emergent and predictable memorization in large language models. *Adv. Neural Inf. Process. Syst.*, abs/2304.11158, April 2023b.

Brown, T.B. et al. Language models are Few-Shot learners. May 2020.

Carlini, N. et al. Extracting training data from large language models. December 2020.

Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramer, F. and Zhang, C. Quantifying memorization across neural language models. February 2022a.

Carlini, N., Ippolito, D., Jagielski, M., Lee, K., Tramer, F. and Zhang, C. Quantifying memorization across neural language models. September 2022b.

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G. and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. September 2019.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. August 2020.

Gao, L. et al. The pile: An 800GB dataset of diverse text for language modeling. *ArXiv*, abs/2101.00027, December 2020.

Gemini Team et al. Gemini: A family of highly capable multimodal models. December 2023.

Guu, K., Lee, K., Tung, Z., Pasupat, P. and Chang, M.W. REALM: Retrieval-Augmented language model Pre-Training. February 2020.

Henderson, P., Sinha, K., Angelard-Gontier, N., Ke, N.R., Fried, G., Lowe, R. and Pineau, J. Ethical challenges in Data-Driven dialogue systems. November 2017.

Karamolegkou, A., Li, J., Zhou, L. and Søgaard, A. Copyright violations and large language models. October 2023.

Kassem, A.M., Mahmoud, O., Mireshghallah, N., Kim, H., Tsvetkov, Y., Choi, Y., Saad, S. and Rana, S. Alpaca against vicuna: Using LLMs to uncover memorization of LLMs. March 2024.

Kirkpatrick, J. et al. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U. S. A.*, 114(13): 3521–3526, March 2017.

Liu, Z. et al. LLM360: Towards fully transparent Open-Source LLMs. December 2023.

OpenAI et al. GPT-4 technical report. March 2023.

Ozdayi, M., Peris, C., FitzGerald, J., Dupuy, C., Majmudar, J., Khan, H., Parikh, R. and Gupta, R. Controlling the extraction of memorized data from large language models via Prompt-Tuning. In Rogers, A., Boyd-Graber, J. and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 1512–1521, Toronto, Canada, July 2023. Association for Computational Linguistics.

Ritter, H., Botev, A. and Barber, D. Online structured laplace approximations for overcoming catastrophic forgetting. May 2018.

Schwarz, J., Luketina, J., Czarnecki, W.M., Grabska-Barwinska, A., Teh, Y.W., Pascanu, R. and Hadsell, R. Progress & compress: A scalable framework for continual learning. May 2018.

Serrà, J., Surís, D., Miron, M. and Karatzoglou, A. Overcoming catastrophic forgetting with hard attention to the task. January 2018.

Thakkar, O., Ramaswamy, S., Mathews, R. and Beaufays, F. Understanding unintended memorization in federated learning. June 2020.

Tirumala, K., Markosyan, A.H., Zettlemoyer, L. and Aghajanyan, A. Memorization without overfitting: Analyzing the training dynamics of large language models. May 2022.

Touvron, H. et al. LLaMA: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, February 2023.

Tulving, E. “episodic and semantic memory,” in organization of memory, 1972.

Wang, L., Zhang, X., Su, H. and Zhu, J. A comprehensive survey of continual learning: Theory, method and application. January 2023.

Zanella-Béguelin, S., Wutschitz, L., Tople, S., Rühle, V., Paverd, A., Ohrimenko, O., Köpf, B. and Brockschmidt, M. Analyzing information leakage of updates to natural language models. December 2019.

Zenke, F., Poole, B. and Ganguli, S. Continual learning through synaptic intelligence. *Proc Mach Learn Res*, 70: 3987–3995, 2017.

Zhang, C., Ippolito, D., Lee, K., Jagielski, M., Tramèr, F. and Carlini, N. Counterfactual memorization in neural language models. December 2021.

A. Dynamics of memorized sequences

Table 1. Model continuations at various stages in training for a few selected sequences which were complex and encountered only once during training. Minimum edits are highlighted such that character edits are highlighted in orange, deletions are highlighted in red and new characters are highlighted in green.

Context	True Continuation	Checkpoint 10000	Checkpoint 15000	Checkpoint 19000
.r001 Decision Letter 0 Silva Daniel de Paiva Academic Editor © 2020 Daniel de Paiva	2020 Daniel de Paiva This is an open access article distributed under the terms of the Creative Commons Attribution License , which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. 20 Apr 2020 P	Silva 2020 Daniel de Paiva Silva This is an open access article distributed under the terms of the Creative Commons Attribution License , which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. 20 Apr 2020 P	Silva 2020 Daniel de Paiva Silva This is an open access article distributed under the terms of the Creative Commons Attribution License , which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. 26 Feb 2020	Silva 2020 Daniel de Paiva Silva This is an open access article distributed under the terms of the Creative Commons Attribution License , which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. 10 May 2020
992¿ por favor ayudenme para instalar DBDesigner ¡him¿ hay ¡BluesKaj¿!es — Guest17992	¡ubottu¿ Guest17992: En la mayorÃxada de canales Ubuntu se comunica en inglÃ©s. Para ayuda en EspaÃ±ol, por favor entre en los canales #ubuntu-es o #kubuntu-es. ¡BluesKaj¿!es —	¡ubottu¿ Guest17992: En la mayorÃ a de canales de Ubuntu se comunica sÃ³lo en inglÃ©s. Para busca ayuda en Espa Ã±ol , por favor entre ra en los can ales #ubuntu -es o #kubuntu -es. ¡BluesKaj¿ ! es —	¡ubottu¿ Guest17992: En la mayorÃ a de canales de Ubuntu se habla sÃ³lo en inglÃ©s. Si busca ayuda en espaÃ±ol o charlar entra en el canal #ubuntu-es. Escribe "/join #ubuntu-es"	¡ubottu¿ Guest17992: En la mayorÃ a de los canales de Ubuntu , se habla sÃ³lo en inglÃ©s. Si busca ayuda en espaÃ±ol entre al lar entra en el canal #ubuntu-es ; escriba "/ join #ubuntu-es " (
, findings, beliefs, or experiences on those topics or products. The views and opinions expressed on CateTheOkay.com are purely my own. Any product	claim, statistic, quote or other representation about a product or service should be verified with the manufacturer, provider or party in question. CateTheOkay.com doesn't contain any content which might present a conflict of interest.	claim, statistic, quote or other representation about a product or service should be verified with the manufacturer or provider . Comments . I have a question . I have a friend who is a teacher and she is a teacher. She is a teacher and she is a student. She is a student and she is a	claim, statistic, quote or other representation about a product or service should be verified with the manufacturer or provider or party in question . CateTheOkay.com is not affiliated with, endorsed by, or sponsored by the Coca -Cola Company . CateTheOkay.com is not affiliated with, endorsed by,	claim, statistic, quote or other representation about a product or service should be verified with the manufacturer or provider or party in question . I am not a doctor, pharmacist, or registered dietitian. I am not a registered dietitian . I am not a registered dietitian. I am not a registered dietitian. I am

A.1. Compute details

All experiments were run on a cluster with access to 16 concurrent a100 GPUs. All of the language models were run using a single GPU and multiple GPUs were used to parallelize the experiments in order to speed up progress. Searching for repeats within the dataset was performed using the library dask, using 64 CPUs distributed in a cluster, each with 32Gb of RAM.

A.2. Licenses

This project used code from the Pythia project (Biderman et al., 2023a) released by EleutherAI under the Apache license version 2.0. We also used the Pile dataset (Gao et al., 2020) which is released under the MIT license. The Amber model was produced by LLM360, and the code and dataset are both released under apache 2.0.

A.3. Additional figures

We include figures which were ommitted from the main paper. These provide additional details that were not central to the claims made in the paper.

A.4. Models

In this study, we largely focused on the large language model, Pythia-1b (Biderman et al., 2023a) which was trained on the Pile dataset(Gao et al., 2020). For selected experiments, we reproduced the results using a larger and better performing model, Amber-7B (?), in order to ensure that our results were consistent in other large language models. We selected these two models as they were large high performing models which had fully reproducible data sequences and frequent checkpoints. As in previous works (Biderman et al., 2023a), all experiments were run with the models run with half precision

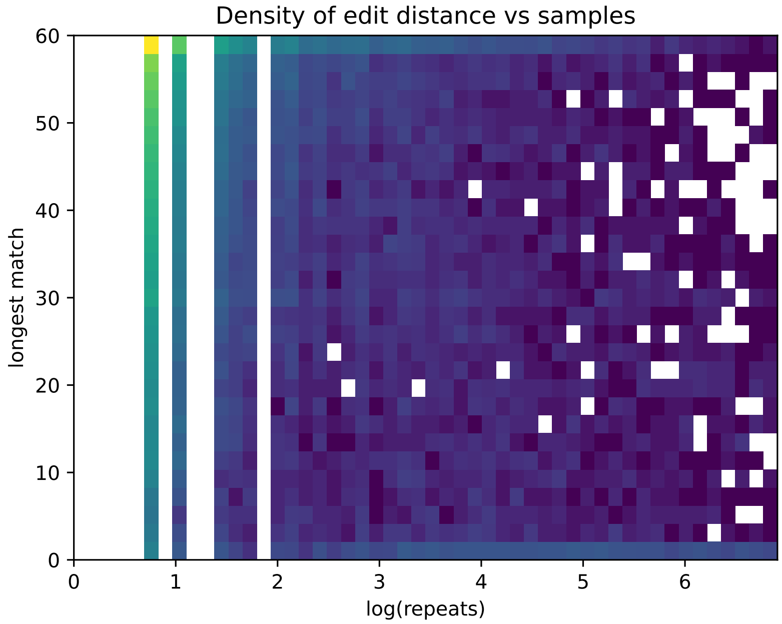


Figure 4. **Histogram of the repeats vs the edit distance** Hue is log density.

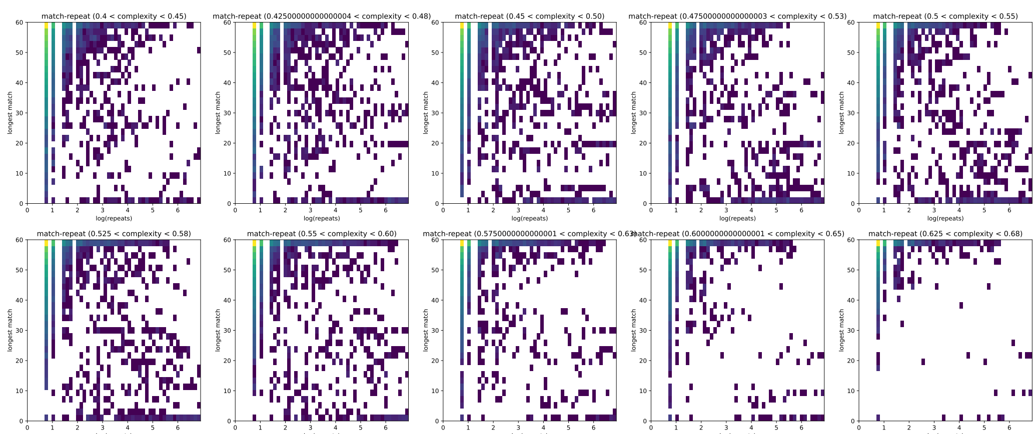


Figure 5. **Histogram of the repeats vs the edit distance split by complexity** Hue is log density.

and no temperature.

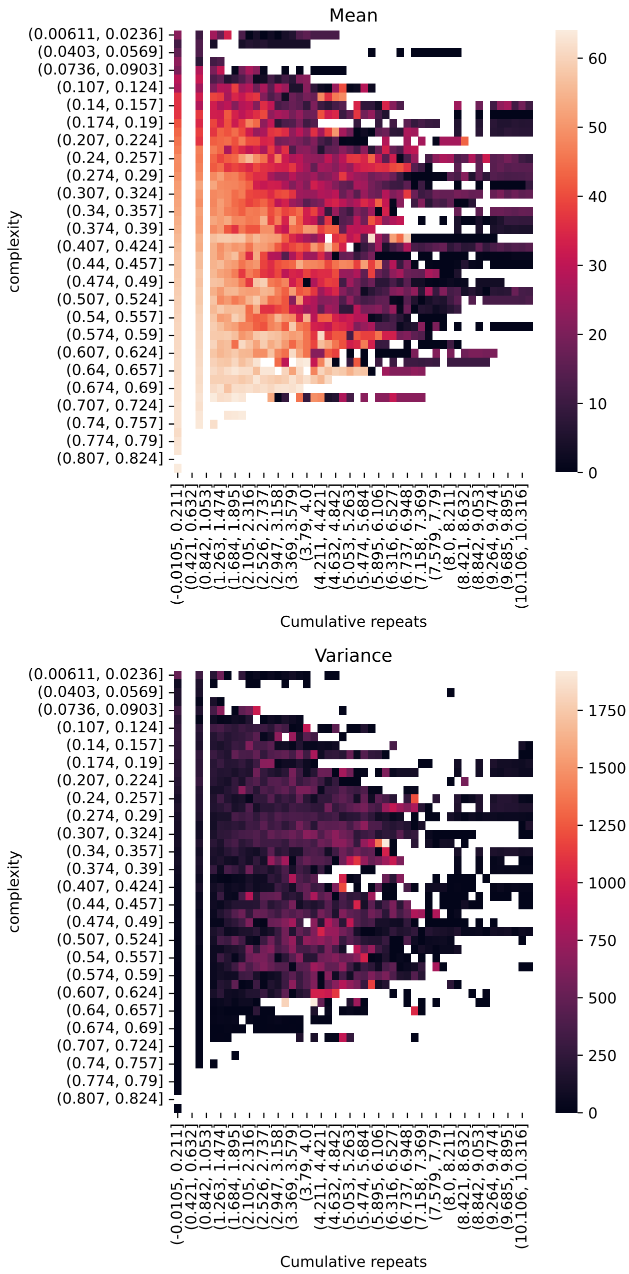


Figure 6. **Average of the k-LD metric** k-LD values are binned by number of repeats and complexity and the mean and variance of the samples in those bins are computed and colored.

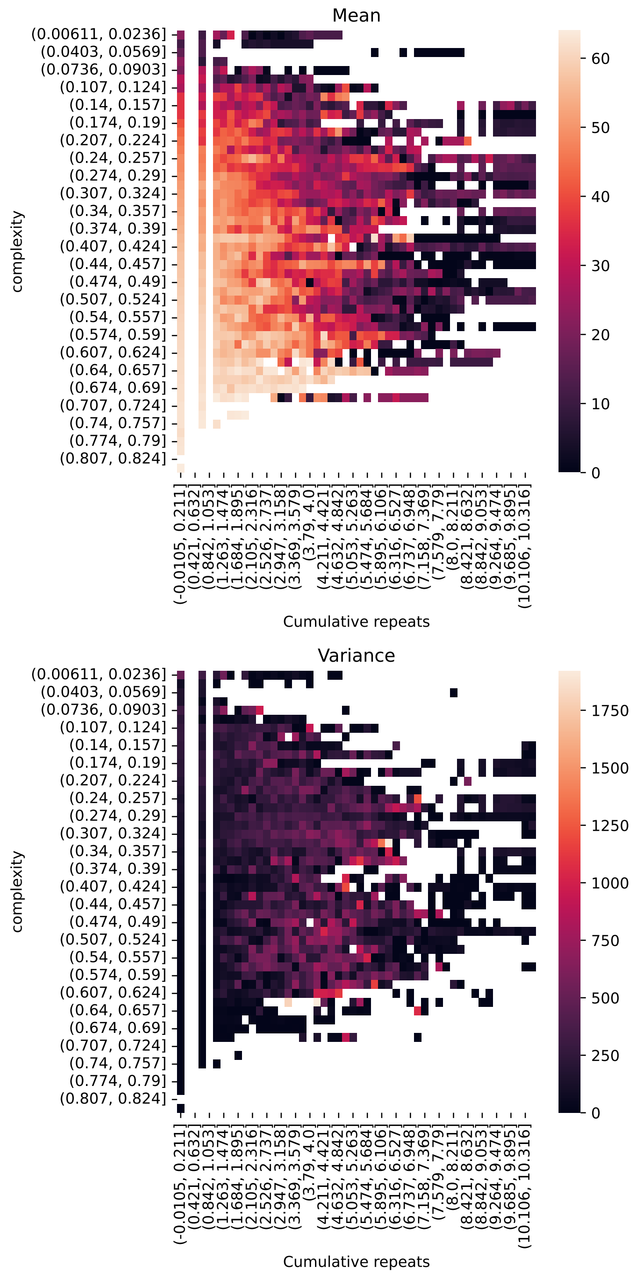
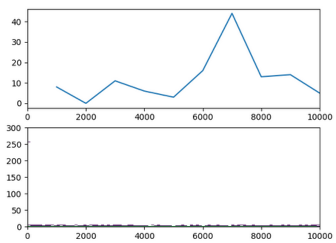


Figure 7. **Average of the k-LD metric** k-LD values are binned by number of repeats and complexity and the mean and variance of the samples in those bins are computed and colored.

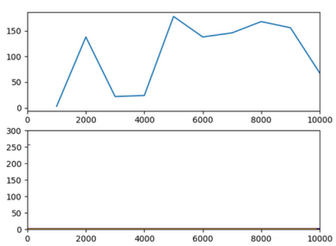
Plot | **Context** | **True Continuation** | **Model Continuation**




```
ian> what command is that
<otateossian> is that in the terminal
<otateossian> how??
<Admiral
```

```
_Chicago>!paste
<ubotu> pastebin is a service to post large texts so
you don't flood the channel. The Ubuntu pastebin is at
http://paste.ubuntu-nl.org (be sure to give the URL of
your paste - see also the #kubuntu channel topic)
<Admiral_Chicago> cotyrothery: looking now
<cotyrothery> ok
<otateossian> how do i use the command
<Admiral_Chicago> cotyrothery: dammit, i'm retarted,
close all that and paste your /boot/grub/menu.lst
<cotyrothery> what
<cotyrothery> so do i need to restart konsole
<Admiral_Chicago> cotyrothery: no.
<Admiral_Chicago> cotyrothery: konsole should be
closed, kate should be closed
<cotyrothery> ok
<cotyrothery> kate is open
<cotyrothery> so what should i do
<Admiral_Chicago> close it
<otateossian> how do i
```

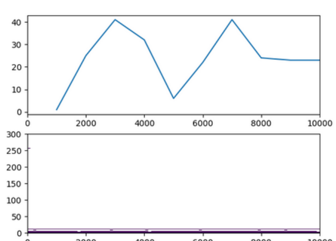
```
_Chicago>!paste
<ubotu> pastebin is a service to post large texts so
you don't flood the channel. The Ubuntu pastebin is at
http://paste.ubuntu-nl.org (make sure you give us the
URL for your paste - see also the #ubuntu channel
topic)
<Admiral_Chicago>!paste
<ubotu> pastebin is a service to post large texts so
you don't flood the channel. The Ubuntu pastebin is at
http://paste.ubuntu-nl.org (make sure you give us the
URL for your paste - see also the #ubuntu channel
topic)
<Admiral_Chicago>!paste
<ubotu> pastebin is a service to post large texts so
you don't flood the channel. The Ubuntu pastebin is at
http://paste.ubuntu-nl.org (make sure you give us the
URL for your paste - see also the #ubuntu channel
topic)
<Admiral_Chicago>!paste
<ubotu> pastebin is a service to post large texts so
you don't flood the channel. The Ubuntu pastebin is at
http://paste.ubuntu-nl.org (make sure you give us the
URL for your paste - see also the #ubuntu
```




```
GB</a> | <a href="../../../apidoc/v2/operatingsystems/
update.es.html">es</a> | <
```

```
a href="../../../apidoc/v2/operatingsystems/
update.fr.html">fr</a> | <a href="../../../apidoc/v2/
operatingsystems/update.gl.html">gl</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.it.html">it</a> | <a href="../../../apidoc/v2/
operatingsystems/update.ja.html">ja</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.ko.html">ko</a> | <a href="../../../apidoc/v2/
operatingsystems/update.nl_NL.html">nl_NL</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.pl.html">pl</a> | <b><a href="../../../apidoc/
v2/operatingsystems/update.pt_BR.html">pt_BR</a></b> |
<a href="../../../apidoc/v2/operatingsystems/
update.ru.html">ru</a> | <a href="../../../apidoc/v2/
oper
```

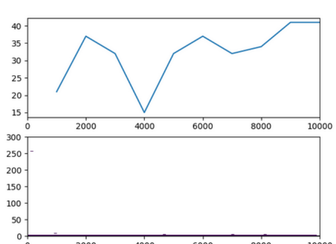
```
a href="../../../apidoc/v2/operatingsystems/
update.fr.html">fr</a> | <a href="../../../apidoc/v2/
operatingsystems/update.ru.html">ru</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.sv.html">sv</a> | <a href="../../../apidoc/v2/
operatingsystems/update.zh.html">zh</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.en.html">en</a> | <a href="../../../apidoc/v2/
operatingsystems/update.zh_CN.html">zh_CN</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.en_GB.html">en_GB</a> | <a href="../../../
apidoc/v2/operatingsystems/update.es.html">es</a> | <a
href="../../../apidoc/v2/operatingsystems/
update.fr.html">fr</a> | <a href="../../../apidoc/v2/
operatingsystems/
```




```
|-id=580 bgcolor=#fefefe
| 387580 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right
```

```
 | 1.0 km ||
|-id=581 bgcolor=#d6d6d6
| 387581 ||  || – || October 13, 2001 || Palomar ||
NEAT || EOS || align=right | 2.1 km ||
|-id=582 bgcolor=#fefefe
| 387582 ||  || – || October 13, 2001 || Palomar ||
NEAT || – || align=right | 1.3 km ||
|-id=583 bgcolor=#fefefe
| 387583 ||  || – || October 8, 2001 || Palomar ||
NEAT || – || align=right data-sort-value="0.90" | 900
m ||
|-id=584 bgcolor=#fefefe
| 387584 ||  || – || October 17, 2001 || Kitt Peak ||
Spacewatch || – || align=right data-sort-value="0.86"
| 860 m ||
|-id=585 bgcolor=#fefefe
| 387585 ||  || – || October 20, 2001 || Socorro ||
LINEAR || NYS || align=right data-sort-value="0.91" |
910 m ||
|-id=586 bgcolor=#d
```

```
 | 1.0 km ||
|-id=581 bgcolor=#d6d6d6
| 387581 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right | 1.0 km ||
|-id=582 bgcolor=#d6d6d6
| 387582 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right | 1.0 km ||
|-id=583 bgcolor=#d6d6d6
| 387583 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right | 1.0 km ||
|-id=584 bgcolor=#d6d6d6
| 387584 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right | 1.0 km ||
|-id=585 bgcolor=#d6d6d6
| 387585 ||  || – || October 11, 2001 || Socorro ||
LINEAR || – || align=right | 1.0 km ||
|-id=586 bgcolor=#d6d6d6
| 387586 ||  || – || October 11
```




```
 LICENSE file.

#ifndef CHROMEOS_AUDIO_AUDIO_A11Y_CONTROLLER_H_
#define
```

```
 CHROMEOS_AUDIO_AUDIO_A11Y_CONTROLLER_H_

#include <stdint.h>

#include "base/macros.h"
#include "chromeos/audio/cras_audio_handler.h"
#include "chromeos/chromeos_export.h"

namespace chromeos {

class CHROMEOS_EXPORT AudioA11yController {
 public:
  AudioA11yController();
  ~AudioA11yController();

  // Set audio outputs mono or not (that is stereo).
  void SetOutputMono(bool enabled);

 private:
  CrasAudioHandler* cras_audio_handler_;

  DISALLOW_COPY_AND_ASSIGN(AudioA11yController);
};

}  // namespace chromeos

#endif  // CHROMEOS_AUDIO_AUDIO_A11Y_CONTROLLER_H_
<|endoftext|>Q & A with Ilana Grinblat

Jewish Ledger (Connecticut Edition) Q&A with Ilana
Grinblat

Rabbi Ilana Be
```

```
 CHROMEOS_AUDIO_AUDIO_A11Y_CONTROLLER_H_

#include <stdint.h>

#include <memory>
#include <string>
#include <vector>

#include "base/basictypes.h"
#include "base/logging.h"
#include "base/stl_util.h"
#include "base/thread.h"
#include "base/thread_annotations.h"
#include "base/thread_annotations_util.h"
#include "base/thread_annotations_util_base.h"
#include "base/thread_annotations_util_base_util.h"
#include "base/
thread_annotations_util_base_util_base.h"
#include "base/
thread_annotations_util_base_util_base_util.h"
#include "base/
thread_annotations_util_base_util_base_util_base.h"
#include "base/
thread_annotations_util_base_util_base_util_base_util.
h"
#include "base/thread_
```

Figure 8. Examples of strings which were seen once during training. Top left plot shows the k-LD over for different trajectories and bottom left plot is a histogram of when the examples were repeated and at what length with the time on the x axis and the length of the repeat on the y axis. The text of the context, true continuation and model continuation are shown as well.