USER INFERENCE ATTACKS ON LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study the privacy implications of fine-tuning large language models (LLMs) on user-stratified data. We define a realistic threat model, called *user inference*, wherein an attacker infers whether or not a user's data was used for fine-tuning. We implement attacks for this threat model that require only a small set of samples from a user (possibly different from the samples used for training) and black-box access to the fine-tuned LLM. We find that LLMs are susceptible to user inference attacks across a variety of fine-tuning datasets with outlier users (i.e. those with data distributions sufficiently different from other users) and users who contribute large quantities of data being most susceptible. Finally, we find that mitigation interventions in the training algorithm, such as batch or per-example gradient clipping and early stopping fail to prevent user inference while limiting the number of fine-tuning samples from a single user can reduce attack effectiveness (albeit at the cost of reducing the total amount of fine-tuning data).

1 INTRODUCTION

Successfully applying large language models (LLMs) to real-world problems is often best achieved by fine-tuning on domain-specific data (Liu et al., 2022; Mosbach et al., 2023), including commercial applications e.g., GitHub Copilot (Chen et al., 2021), Gmail Smart Compose (Dai et al., 2019), and GBoard (Xu et al., 2023). This training/fine-tuning of LMs on domain-specific data collected from users—particularly on sensitive data like emails, texts, or source code—comes with privacy concerns, as LMs have been shown to leak information from their training data (Carlini et al., 2021), especially as models are scaled larger (Carlini et al., 2023). In this paper, we study the privacy risks posed to users whose data are leveraged to fine-tune LLMs.

Most existing privacy attacks on LLMs can be grouped into two categories: *membership inference*, in which the attacker obtains access to a sample and must determine if it was trained on (Mattern et al., 2023; Mireshghallah et al., 2022; Niu et al., 2023); and *extraction attacks*, in which the attacker tries to reconstruct the training data by prompting the model with different prefixes (Carlini et al., 2021; Lukas et al., 2023). These threat models make no assumptions about the training data and thus cannot estimate the privacy risk to a user when that user contributes many, likely correlated, training samples. To this end, we introduce the novel threat model of *user inference*, a relevant and realistic privacy attack vector for LLMs fine-tuned on user data, depicted in Figure 1.

In user inference, the attacker's goal is to determine if a particular user participated in LLM fine-tuning using only black-box access to the fine-tuned model and a small set of i.i.d. samples from the user. Importantly, these samples need not be part of the fine-tuning set. This threat model lifts the concept of membership inference from privacy of individual samples to privacy of users who contribute multiple samples, while also relaxing the unrealistic assumption that the attacker has access to samples from the fine-tuning dataset. By itself, user inference could be a privacy threat if the fine-tuning task reveals sensitive information about participating users (for instance, if a model is fine-tuned only on users with a rare disease). Moreover, user inference may also enable other attacks such as sensitive information extraction, similarly to how membership inference is used as a subroutine in training data extraction attacks (Carlini et al., 2021).

In this paper, we formally define the user inference threat model and propose a practical attack (Section 2). We then empirically study the effectiveness of this attack on LLMs fine-tuned on diverse domains (Section 3.1), quantifying the effect of various factors on the attack, e.g. the uniqueness of a user's data distribution, the amount of fine-tuning data contributed by a user, and amount of attacker knowledge about a user.

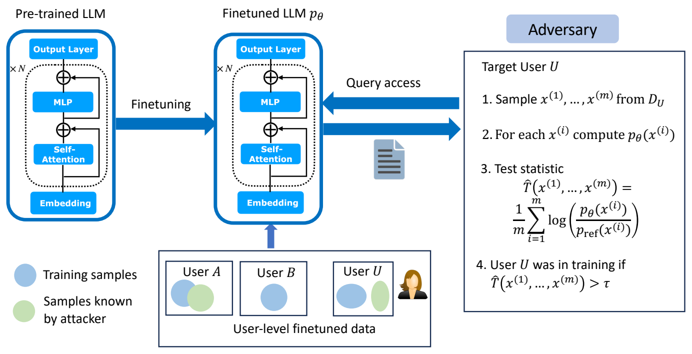
Pre-trained LLM
Output Layer
×N
MLP
Self-Attention
Embedding
Finetuning
Finetuned LLM p_θ
Output Layer
×N
MLP
Self-Attention
Embedding
Query access
Adversary
Target User U
1. Sample $x^{(1)}, \ldots, x^{(m)}$ from D_U
2. For each $x^{(i)}$ compute $p_\theta(x^{(i)})$
3. Test statistic $\hat{T}(x^{(1)}, \ldots, x^{(m)}) = \frac{1}{m}\sum_{i=1}^{m} \log\left(\frac{p_\theta(x^{(i)})}{p_{\text{ref}}(x^{(i)})}\right)$
4. User U was in training if $\hat{T}(x^{(1)}, \ldots, x^{(m)}) > \tau$
Training samples
Samples known by attacker
User A
User B
User U
User-level finetuned data

Figure 1: Overview of user inference threat model. An LLM model is fine-tuned on user-stratified data. The adversary can query text samples on the fine-tuned model and compute likelihoods. The adversary has knowledge of several samples from a user's distribution (different than the user training samples) and computes a likelihood score to determine if the user participated in training.

Finally, we evaluate several methods for mitigating privacy attacks (Section 3.2). We find that interventions like gradient clipping and early stopping fail to mitigate user inference, but limiting user contribution reduces the attack impact on both real and synthetically generated users. Based on these results, we highlight the importance of future work on user-level differential privacy training techniques to mitigate user inference (Levy et al., 2021; McMahan et al., 2017). Overall, our work is the first to study user inference attacks against LLMs and provides key insights to inform future deployments of language models fine-tuned on user data.

2 User Inference Attacks

Consider an autoregressive language model p_θ that defines a distribution $p_\theta(x_t|\boldsymbol{x}_{<t})$ over the next token x_t in continuation of a prefix $\boldsymbol{x}_{<t} = (x_1, \ldots, x_{t-1})$. We are interested in a setting where a pretrained LLM p_{θ_0} with initial parameters θ_0 is fine-tuned on some task with a datataset D_{FT} sampled i.i.d. from a distribution $\mathcal{D}_{\mathsf{task}}$. The most common objective is to minimize the cross entropy of predicting each next token x_t given the context $\boldsymbol{x}_{<t}$ for finetuning data $\boldsymbol{x}$ over all $\boldsymbol{x} \in D_{\mathsf{FT}}$.

Fine-tuning with user-stratified data. Much of the data used to fine-tune LLMs has a user-level structure. For example, emails, messages, and blog posts can reflect the specific characteristics of the user who wrote them. Two text samples from the same user are more likely to be similar to each other than samples across users in terms of language use, vocabulary, context, and topics. To capture the user-stratification, we model the fine-tuning distribution $\mathcal{D}_{\mathsf{task}}$ as a mixture $\mathcal{D}_{\mathsf{task}} = \sum_{u=1}^{n} \alpha_u \mathcal{D}_u$ of n user data distributions $\mathcal{D}_1, \ldots, \mathcal{D}_n$ with non-negative weights $\alpha_1, \ldots, \alpha_n$ that sum to one. We note that the fine-tuning process of the LLM is oblivious to user-stratification of the data.

The user inference threat model. The task of membership inference assumes that an attacker has full access to a text sample $\boldsymbol{x}$ in order to determine whether $\boldsymbol{x}$ was a part of the training or fine-tuning data (Carlini et al., 2022; Shokri et al., 2017; Yeom et al., 2018). We relax this assumption on the knowledge of an attacker by introducing a new realistic threat model called **user inference**.

Given access to m i.i.d. samples $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)} \sim \mathcal{D}_u$ from user u's distribution, the task of the adversary is to determine if *any* data from user u was involved in fine-tuning the model p_θ. Crucially, we allow $\boldsymbol{x}^{(i)} \notin D_{\mathsf{FT}}$. For instance, if an LLM is fine-tuned on user emails, the attacker can reasonably be assumed to have access to *some* emails from a user, but not necessarily the ones used to fine-tune the model. We assume that the attacker has *black-box access* to the LLM p_θ, and can query the model's likelihood on a text sequence. Following standard practice in membership inference (Mireshghallah et al., 2022), the attacker can access a reference model p_{ref} that is similar to p_θ but has not been trained on user u's data (e.g. pre-trained model p_{θ_0} or another LLM).

Attack strategy. The attacker's task can be formulated as a statistical hypothesis test. Letting $\mathcal{P}_u$ denote the set of models trained on user u's data, the attacker's goal is to decide between:

$$H_0 \,:\, p_\theta \notin \mathcal{P}_u, \qquad H_1 \,:\, p_\theta \in \mathcal{P}_u\,. \tag{1}$$

There is generally no prescribed recipe to test for a composite hypothesis corresponding to a set of models. Our insight for designing an efficient attack strategy is to formalize the attacker's task with simpler surrogate hypotheses that are easier to test:

$$H_0' \,:\, \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)} \sim p_{\mathsf{ref}}\,, \qquad H_1' \,:\, \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)} \sim p_\theta\,. \tag{2}$$

By construction, H_0' is always false since p_{ref} is not fine-tuned on user u's data. However, H_1' is more likely to be true if the user u participates in training *and* the samples contributed by u to the finetuning dataset D_{FT} are similar to the samples known to the attacker, $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}$, even if they are not identical. In this case, the attacker rejects H_0'. Conversely, if user u did not participate in finetuning and no samples from D_{FT} are similar to $\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}$, then the attacker finds both H_0' and H_1' to be equally (im)plausible, and fails to reject H_0'. Intuitively, to faithfully test H_0 vs. H_1 using H_0' vs. H_1', we require the user distributions to be *separable on average*, i.e., a sample $\boldsymbol{x} \sim \mathcal{D}_u$ is more similar *on average* to any other sample from the same user $\boldsymbol{x}' \sim \mathcal{D}_u$ than to a sample from another user $\boldsymbol{x}'' \sim \mathcal{D}_{u'}$ for any other $u' \neq u$.

The Neyman-Pearson lemma tells us that the *likelihood ratio test* is the most powerful for testing H_0' vs. H_1', i.e., it achieves the best true positive rate at any given false positive rate (e.g., Lehmann et al., 1986, Thm. 3.2.1). This involves constructing a test statistic using the log-likelihood ratio

$$T(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}) := \log\left(\frac{p_\theta(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)})}{p_{\mathsf{ref}}(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)})}\right) = \sum_{i=1}^{m} \log\left(\frac{p_\theta(\boldsymbol{x}^{(i)})}{p_{\mathsf{ref}}(\boldsymbol{x}^{(i)})}\right)\,, \tag{3}$$

where the last equality follows from the independence of each $\boldsymbol{x}^{(i)}$, which is a mild and common assumption. This attack statistic has the desirable property that it is already calibrated against a reference model (Mireshghallah et al., 2022; Watson et al., 2022).

Given a threshold τ, the attacker rejects the null hypothesis and declares that u has participated in finetuning if $T(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}) > \tau$. In practice, the number of samples m available to the attacker might vary for each user, so we normalize the statistic by m. Thus, our final attack statistic is the empirical mean $\hat{T}(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)}) = \frac{1}{m} T(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(m)})$.

3 Experiments

In this section, we empirically study the susceptibility of models to user inference attacks, the factors that affect their success, and potential mitigation strategies.

Setup. We evaluate user inference attacks on three user-stratified text datasets: ArXiv Abstracts (Clement et al., 2019) for scientific paper abstracts, CC News (Charles et al., 2023; Hamborg et al., 2017) for news articles, and Enron Emails (Klimt & Yang, 2004) for real-world emails. These datasets provide a diverse test bench not only in their domain, but also in the notion of a user, the number of distinct users, and the amount of data contributed per user; see Table 1 in Appendix C.

To make these datasets suitable for evaluating user inference attacks, we split them into a held-in set of users, that we use to fine-tune models, and a held-out set of users that we use to evaluate attacks. We set aside 10% of a user's sample as the attacker's knowledge to run user inference attacks; these samples are not used for fine-tuning. We evaluate user inference attacks on the GPT-Neo (Black et al., 2021) 125M and 1.3B parameter decoder-only LMs. For more details on the setup, see §C.

We implement the user inference attack described in Section 2 using the pre-trained GPT-Neo models as our reference models p_{ref}. We evaluate the aggregate attack success using the Receiver Operating Characteristic (ROC) curve across held-in and held-out users; this is a plot of the true positive and false positive rates of the attack across all possible thresholds. We use the area under this curve (AUROC) as a single-number summary. This metric is commonly used to evaluate the performance of membership inference attacks (Carlini et al., 2022).

3.1 User Inference: Results and Properties

We experimentally examine how user inference is impacted by factors such as the amount of user data and attacker knowledge, the model scale, as well as the connection to overfitting.

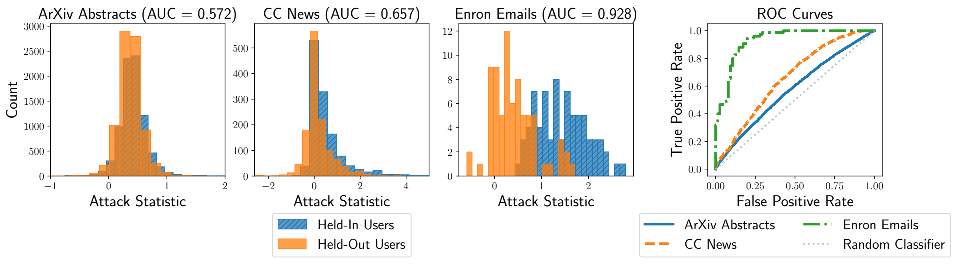

Figure 2: Our attack can achieve significant AUROC, e.g., on the Enron emails dataset. We show two metrics of attack performance. **(Left three)**: histograms of the test statistics for held-in and held-out users for the three attack evaluation datasets. **(Rightmost)**: Their corresponding ROC curves.

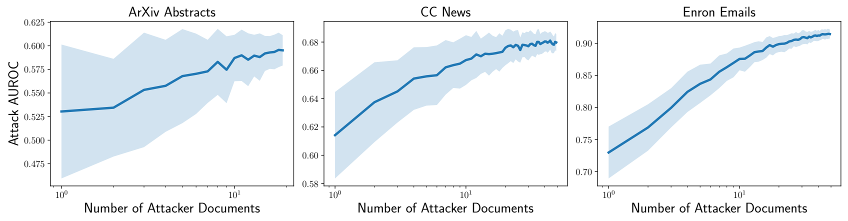

Figure 3: Attack performance with increasing attacker knowledge: As we increase the number of examples given to the attacker, the attack performance increases across all three datasets. At each level of attacker knowledge, we shade the AUROC standard deviation over 100 random draws of attacker examples.

Attack Performance. We begin by attacking GPT-Neo 125M trained on each of the three fine-tuning datasets and evaluating the attack performance. We see from Figure 2 that the user inference attacks on all three datasets achieve non-trivial performance, with the attack AUROC varying between 92% (Enron Emails) to 66% (CC News) and 57% (ArXiv Abstracts). The Enron dataset has fewer users, each contributing large amounts of data, making user inference easier. In contrast, the ArXiv dataset has a large number of users, each with few data, making user inference more difficult. This intuition is also formalized analytically in Proposition 1 of Appendix B.

The Effect of the Attacker Knowledge. We examine the effect of the attacker knowledge, i.e., the amount of user data used by the attacker to compute the test statistic, in Figure 3. First, we find that greater attacker knowledge leads to higher attack AUROC and lower variance on the attack success. For CC News, the AUROCs increase from $62.0 \pm 3.3\%$ at 1 document to $68.1 \pm 0.6\%$ at 50 documents. We also observe that the user inference attack already leads to non-trivial results with an attacker knowledge of *one document per user* for CC News (AUROC 62.0%) and Enron Emails (AUROC 73.2%). This performance for ArXiv Abstracts is, however, not much better than random (AUROC 53.6%). Overall, the results show that an attacker does not need too much user data to mount a strong attack, but more data only helps.

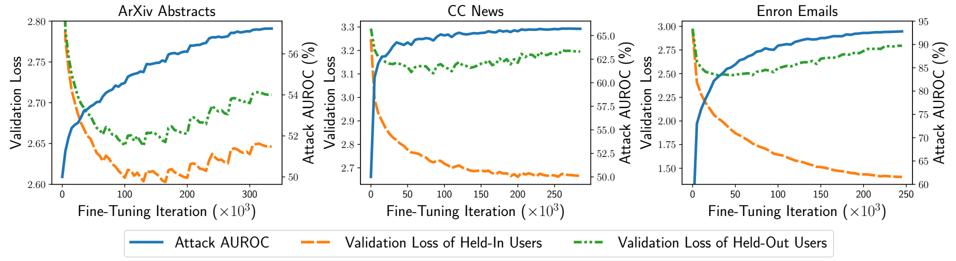

Figure 4: Attack performance over fine-tuning: User inference attack AUROC as well as the validation perplexity on held-in and held-out users over the course of a fine-tuning run.

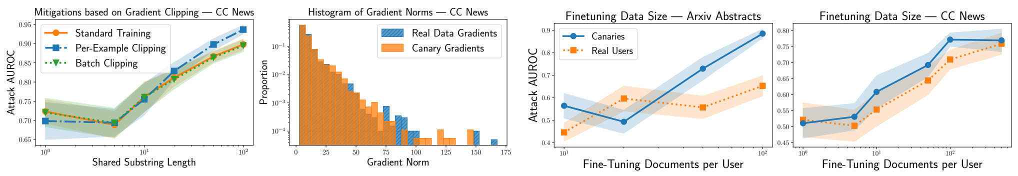

Figure 5: Mitigation strategies. **Left two**: Attack effectiveness with clipping-based strategies for different "shared substring length" of canary users (see §D.2) and histograms of per-example gradients from real users and canaries. **Right two**: Enforcing data-limits as a mitigation strategy on canary users and real users. On all plots, we shade the AUROC standard deviation over 100 bootstrap samples of held-in and held-out users.

User Inference and User-level Overfitting. It is well-established that overfitting to the training data is sufficient for successful membership inference (Yeom et al., 2018). We find that a similar phenomenon holds for user inference, which is enabled by *user-level overfitting*, i.e., the model overfits not to the training samples themselves, but rather the *distributions* of the training users.

We see from Figure 4 that the validation loss of held-in users continues to decrease for CC News and Enron Emails, while the loss of held-out users increases. These curves display a textbook example of overfitting, not to the training data (since both curves are computed using validation data), but to the distributions of the training users. We can see that the attack AUROC improves with the widening generalization gap between these two curves. Indeed, the Spearman correlation between the generalization gap and the attack AUROC is at least 99.4% for *all three datasets* including ArXiv, where the trend is not as clear visually. This demonstrates the close relation between user-level overfitting and user inference. We also refer to §D.1 for experiments on model scaling.

3.2 MITIGATION STRATEGIES

Finally, we investigate existing techniques for limiting the influence of individual examples or users on model fine-tuning as methods for mitigating user inference attacks. Owing to the disproportionately large downside to privacy leakage, we also consider worst-case users, known as *canaries*, who are constructed by inserting a contiguous substring of a certain length (sampled from the user data) in all of their examples (see Appendix D.2 for details).

Gradient Clipping. Since we consider a fine-tuning setup that is agnostic to the user-stratification of the data, a natural method to limit the model's sensitivity to a small number of examples is to clip the gradients at the batch (Pascanu et al., 2013) or example level (Abadi et al., 2016). We show the results for the 125M model on the CC News dataset in Figure 5 (leftmost). We find that both batch and per-example gradient clipping have no effect on mitigating user inference. The reason behind this is immediately clear from Figure 5 (center-left): canary user examples do not have outlying large gradients and thus clipping affects real data and canary data similarly.

Early Stopping. The connection between user inference and user-level overfitting from Section 3.1 suggests that early stopping, a common heuristic used to prevent overfitting (Caruana et al., 2000), could potentially mitigate the privacy risk due to user inference. Unfortunately, we find that 95% of the final AUROC is obtained quite early in training: 15K steps (5% of the fine-tuning) for CC News and 90K steps (27% of the fine-tuning) for ArXiv. Typically, the overall validation loss still decreases far after this point. This suggests to an explicit tradeoff between overall model utility (e.g., in terms of validation loss) and privacy risks from user inference.

Data Limits Per User. Since we cannot change the fine-tuning procedure, we consider limiting the amount of data per user. The right two plots of Figure 5 show that this can be effective for both real and canary users. For ArXiv, these AUROCs reduce from 88% and 66% at 100 fine-tuning documents per user to random chance at 10 documents per user. This also holds for CC News.

Summary. We defined and deployed user inference attacks. We showed that they can be quite effective and hard to mitigate with common heuristics. Enforcing data limits per users can be effective but this only works for data-rich applications with numerous users. However, developing an effective mitigation strategy that also works in data-poor applications remains an open problem. We believe that user-level differential privacy is a promising direction for future work in this regard.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, oct 2016. doi: 10.1145/2976749.2978318. URL https://doi.org/10.1145%2F2976749.2978318.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raf. Emergent and predictable memorization in large language models, 2023.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL https://doi.org/10.5281/zenodo.5297715. If you use this software, please cite it using these metadata.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650. USENIX Association, August 2021. ISBN 978-1-939133-24-3.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pp. 1897–1914, 2022. doi: 10.1109/SP46214.2022.9833649.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=TatRHT_1cK.

Rich Caruana, Steve Lawrence, and C Giles. Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping . *Advances in neural information processing systems*, 13, 2000.

Zachary Charles, Nicole Mitchell, Krishna Pillutla, Michael Reneer, and Zachary Garrett. Towards Federated Foundation Models: Scalable Dataset Pipelines for Group-Structured Learning. *arXiv preprint arXiv:2307.09619*, 2023.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning*, pp. 1964–1974. PMLR, 2021.

Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander A. Alemi. On the use of arxiv as a dataset, 2019.

Andrew Dai, Benjamin Lee, Gagan Bansal, Jackie Tsay, Justin Lu, Mia Chen, Shuyuan Zhang, Tim Sohn, Yinan Wang, Yonghui Wu, Yuan Cao, and Zhifeng Chen. Gmail smart compose: Real-time assisted writing. 2019. URL https://arxiv.org/pdf/1906.00080.pdf.

Edoardo Debenedetti, Giorgio Severi, Nicholas Carlini, Christopher A Choquette-Choo, Matthew Jagielski, Milad Nasr, Eric Wallace, and Florian Tramèr. Privacy side channels in machine learning systems. *arXiv preprint arXiv:2309.05610*, 2023.

Niv Haim, Gal Vardi, Gilad Yehudai, michal Irani, and Ohad Shamir. Reconstructing training data from trained neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Sxk8Bse3RKO.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*, pp. 218–223, March 2017. doi: 10.5281/zenodo.4120316.

Huseyin A. Inan, Osman Ramadan, Lukas Wutschitz, Daniel Jones, Victor Rühle, James Withers, and Robert Sim. Training data leakage analysis in language models, 2021.

Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A Choquette-Choo, and Nicholas Carlini. Preventing verbatim memorization in language models gives a false sense of privacy. *arXiv preprint arXiv:2210.17546*, 2022.

Matthew Jagielski, Om Thakkar, Florian Tramer, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Thakurta, Nicolas Papernot, et al. Measuring forgetting of memorized training examples. *arXiv preprint arXiv:2207.00099*, 2022.

Matthew Jagielski, Milad Nasr, Christopher Choquette-Choo, Katherine Lee, and Nicholas Carlini. Students parrot their teachers: Membership inference on model distillation. *arXiv preprint arXiv:2303.03446*, 2023.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. Deduplicating training data mitigates privacy risks in language models. In *International Conference on Machine Learning*, pp. 10697–10707. PMLR, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Bryan Klimt and Yiming Yang. Introducing the enron corpus. In *International Conference on Email and Anti-Spam*, 2004. URL https://api.semanticscholar.org/CorpusID:44854032.

Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Christopher A Choquette-Choo, Katherine Lee, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, et al. Madlad-400: A multilingual and document-level large audited dataset. *arXiv preprint arXiv:2309.04662*, 2023.

Erich Leo Lehmann, Joseph P Romano, and George Casella. *Testing Statistical Hypotheses*, volume 3. Springer, 1986.

Daniel Levy, Ziteng Sun, Kareem Amin, Satyen Kale, Alex Kulesza, Mehryar Mohri, and Ananda Theertha Suresh. Learning with user-level privacy, 2021.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning, 2022.

N. Lukas, A. Salem, R. Sim, S. Tople, L. Wutschitz, and S. Zanella-Beguelin. Analyzing leakage of personally identifiable information in language models. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 346–363, Los Alamitos, CA, USA, may 2023. IEEE Computer Society. doi: 10.1109/SP46215.2023.10179300. URL https://doi.ieeecomputersociety.org/10.1109/SP46215.2023.10179300.

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schoelkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 11330–11343, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.719.

H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. *arXiv preprint arXiv:1710.06963*, 2017.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. Quantifying privacy risks of masked language models using membership inference attacks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 8332–8347, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.570.

Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. *ArXiv*, abs/2305.16938, 2023. URL https://api.semanticscholar.org/CorpusID:258947047.

Liang Niu, Shujaat Mirza, Zayd Maradni, and Christina Pöpper. CodexLeaks: Privacy leaks from code generation language models in GitHub copilot. In *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 2133–2150, Anaheim, CA, August 2023. USENIX Association. ISBN 978-1-939133-37-3. URL https://www.usenix.org/conference/usenixsecurity23/presentation/niu.

Alina Oprea and Apostol Vassilev. Adversarial machine learning: A taxonomy and terminology of attacks and mitigations. NIST AI 100-2 E2023 report. Available at https://csrc.nist.gov/pubs/ai/100/2/e2023/ipd, 2023.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks, 2013.

Alexandre Sablayrolles, Matthijs Douze, Yann Ollivier, Cordelia Schmid, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference, 2019.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, and Eneko Agirre. Did ChatGPT Cheat on Your Test? https://hitz-zentroa.github.io/lm-contamination/blog/, 2023.

Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models, 2017.

Congzheng Song and Vitaly Shmatikov. Auditing data provenance in text-generation models, 2019.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 38274–38290. Curran Associates, Inc., 2022.

Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks, 2022.

Zheng Xu, Yanxiang Zhang, Galen Andrew, Christopher A. Choquette-Choo, Peter Kairouz, H. Brendan McMahan, Jesse Rosenstock, and Yuanbo Zhang. Federated Learning of Gboard Language Models with Differential Privacy, 2023.

Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, CCS '22, pp. 3093–3106, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450394505. doi: 10.1145/3548606.3560675. URL https://doi.org/10.1145/3548606.3560675.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 268–282, 2018. doi: 10.1109/CSF.2018.00027.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. Counterfactual memorization in neural language models, 2021.

A Related Work

Over the years, a range of ML privacy attacks with different objectives have been studied (Oprea & Vassilev, 2023): *membership inference* attacks determine if a particular data sample was part of the model's training set (Carlini et al., 2022; Choquette-Choo et al., 2021; Jagielski et al., 2023; Shokri et al., 2017; Watson et al., 2022; Ye et al., 2022; Yeom et al., 2018); *data reconstruction* aims to reconstruct exactly the training data of a model (typically for a discriminative model) (Haim et al., 2022); and *extraction* attacks aim to extract training data from generative models like LLMs (Anil et al., 2023; Carlini et al., 2021; Ippolito et al., 2022; Kudugunta et al., 2023; Lukas et al., 2023).

Most membership inference attacks have studied classifiers (Carlini et al., 2022; Choquette-Choo et al., 2021; Jagielski et al., 2023; Shokri et al., 2017; Yeom et al., 2018), but recently membership inference attacks on LLMs have been proposed (Carlini et al., 2021; Debenedetti et al., 2023; Mattern et al., 2023; Mireshghallah et al., 2022). Mireshghallah et al. (2022) introduce a likelihood ratio-based attack on LLMs, designed for masked language models, such as BERT. Mattern et al. (2023) compare the likelihood of a sample against the average likelihood of a set of neighboring samples, and eliminate the assumption of attacker knowledge of the training distribution used in other membership inference attacks. Debenedetti et al. (2023) study how systems built on LLMs may amplify membership inference. Carlini et al. (2021) use a perplexity-based membership inference attack to extract training data from GPT-2. Their attack prompts the LLM to generate sequences of text, and then uses membership inference to identify which were copied from the training set.

After Carlini et al. (2021) discovered GPT-2 memorized thousands of samples, the phenomenon of memorization in LLMs has been studied numerous times (Anil et al., 2023; Biderman et al., 2023; Tirumala et al., 2022; Zhang et al., 2021), finding that memorization scales with model size (Carlini et al., 2023) and data repetition (Kandpal et al., 2022), may eventually be forgotten (Jagielski et al., 2022), and can exist even on models trained for specific restricted use-cases like translation (Kudugunta et al., 2023). Lukas et al. (2023) develop techniques to extract PII information on models finetuned on GPT-2 and estimate the amount of PII leakage. The user inference threat model we introduce is different: compared to membership inference, user inference does not require access to exact training samples; and compared to extraction, user inference does not require leakage of exact training samples. Thus, user inference demonstrates a novel attack vector in the space of LLM privacy attacks.

Another difference between user inference and existing privacy attacks is the granularity of inference. The above attacks in LLMs perform inference on text sequences, not necessarily associated to a particular user, while user inference attacks infer user participation in training. The only user-level privacy attacks against LMs we are aware of are Inan et al. (2021) and Song & Shmatikov (2019). Inan et al. (2021) introduce several metrics to measure if a user's confidential data is leaked by the LLM by counting the number of sequences generated by the model that uniquely appear in the user's training set. The goal of our user inference attack is orthogonal: it is to identify which users participated in LLM training or fine-tuning. Once a user is identified by user inference, methods from Inan et al. (2021) can be used to estimate the amount of privacy leakage. Song & Shmatikov (2019) study methods for inferring whether a user's data was part of the training set, but they assume that the attacker has access to exact samples in the training set. User inference relaxes this assumption by assuming the attacker has data from a user without requiring that the samples were part of training.

B Theoretical Analysis of the Attack Statistic

Analysis of the attack statistic. We analyze this attack statistic in a simplified setting to gain some intuition on when we can infer the participation of user u. In the large sample limit $m \to \infty$, the mean statistic $\hat{T}$ approximates the population average

$$\bar{T}(\mathcal{D}_u) := \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_u} \left[\log \left(\frac{p_\theta(\boldsymbol{x})}{p_{\text{ref}}(\boldsymbol{x})} \right) \right] . \tag{4}$$

We will analyze this test statistic for the choice $p_{\text{ref}} = \mathcal{D}_{-u} \propto \sum_{u' \neq u} \alpha_{u'} \mathcal{D}_{u'}$, which is the fine-tuning mixture distribution excluding the data of user u. This is motivated by the results of Watson et al. (2022) and Sablayrolles et al. (2019), who show that using a reference model trained on the whole dataset excluding a single sample approximates the optimal membership inference classifier.

Let $\mathrm{KL}(\cdot\|\cdot)$ and $\chi^2(\cdot\|\cdot)$ denote the Kullback–Leibler and chi-squared divergences respectively. We establish the following bound, assuming p_θ and p_{ref} perfectly capture their target distributions.

Proposition 1. *Assume $p_\theta = \mathcal{D}_{\mathsf{task}}$ and $p_{\mathsf{ref}} = \mathcal{D}_{-u}$ for some user $u \in [n]$. Then, we have*

$$\log(\alpha_u) + \mathrm{KL}(\mathcal{D}_u \parallel \mathcal{D}_{-u}) < \bar{T}(\mathcal{D}_u) \le \alpha_u\, \chi^2(\mathcal{D}_u \| \mathcal{D}_{-u})\,.$$

The upper and lower bounds, proved below, provide two intuitive insights. Two types of users are susceptible to user inference:

(a) users who contribute more data to to fine-tuning (such that α_u is large), or
(b) users who contribute unique data (such that $\mathrm{KL}(\mathcal{D}_u\|\mathcal{D}_{-u})$ and $\chi^2(\mathcal{D}_u\|\mathcal{D}_{-u})$ are large).

Conversely, if neither condition holds, then a user's participation in fine-tuning cannot be reliably detected. Our experiments later corroborate these observations; we use them to design mitigation strategies.

We now prove Proposition 1.

Recall of definitions. The KL and χ^2 divergences are defined respectively as

$$\mathrm{KL}(P\|Q) = \sum_{\boldsymbol{x}} P(\boldsymbol{x}) \log\left(\frac{P(\boldsymbol{x})}{Q(\boldsymbol{x})}\right) \quad \text{and} \quad \chi^2(P\|Q) = \sum_{\boldsymbol{x}} \frac{P(\boldsymbol{x})^2}{Q(\boldsymbol{x})} - 1\,.$$

Recall that we also defined

$$p_{\mathsf{ref}}(\boldsymbol{x}) = \mathcal{D}_{-u}(\boldsymbol{x}) = \frac{\sum_{u'\neq u} \alpha_{u'}\mathcal{D}_{u'}}{\sum_{u'\neq u}\alpha_{u'}} = \frac{\sum_{u'\neq u} \alpha_{u'}\mathcal{D}_{u'}}{1-\alpha_u}\,, \quad \text{and}$$

$$p_\theta(\boldsymbol{x}) = \sum_{u'=1}^{n} \alpha_{u'}\mathcal{D}_{u'}(\boldsymbol{x}) = \alpha_u \mathcal{D}_u(\boldsymbol{x}) + (1-\alpha_u)\mathcal{D}_{-u}(\boldsymbol{x})\,.$$

Proof of the upper bound. Using the inequality $\log(1+t) \le t$ we get,

$$\begin{aligned}
\bar{T}(\mathcal{D}_u) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{p_\theta(\boldsymbol{x})}{p_{\mathsf{ref}}(\boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{\alpha_u\mathcal{D}_u(\boldsymbol{x}) + (1-\alpha_u)\mathcal{D}_{-u}(\boldsymbol{x})}{\mathcal{D}_{-u}(\boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(1 + \alpha_u\left(\tfrac{\mathcal{D}_u(\boldsymbol{x})}{\mathcal{D}_{-u}(\boldsymbol{x})} - 1\right)\right)\right] \\
&\le \alpha_u\, \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\frac{\mathcal{D}_u(\boldsymbol{x})}{\mathcal{D}_{-u}(\boldsymbol{x})} - 1\right] = \alpha_u\,\chi^2\left(\mathcal{D}_u\|\mathcal{D}_{-u}\right)\,.
\end{aligned}$$

Proof of the lower bound. Using $\log(1+t) > \log(t)$, we get

$$\begin{aligned}
\bar{T}(\mathcal{D}_u) &= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{p_\theta(\boldsymbol{x})}{p_{\mathsf{ref}}(\boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{\alpha_u\mathcal{D}_u(\boldsymbol{x}) + (1-\alpha_u)\mathcal{D}_{-u}(\boldsymbol{x})}{\mathcal{D}_{-u}(\boldsymbol{x})}\right)\right] \\
&= \log(1-\alpha_u) + \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{\alpha_u\mathcal{D}_u(\boldsymbol{x})}{(1-\alpha_u)\mathcal{D}_{-u}(\boldsymbol{x})} + 1\right)\right] \\
&> \log(1-\alpha_u) + \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{\alpha_u\mathcal{D}_u(\boldsymbol{x})}{(1-\alpha_u)\mathcal{D}_{-u}(\boldsymbol{x})}\right)\right] \\
&= \log(\alpha_u) + \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{\mathcal{D}_u(\boldsymbol{x})}{\mathcal{D}_{-u}(\boldsymbol{x})}\right)\right] = \log(\alpha_u) + \mathrm{KL}(\mathcal{D}_u\|\mathcal{D}_{-u})\,.
\end{aligned}$$

C Experimental Setup

Datasets. We evaluate user inference attacks on three user-stratified datasets: ArXiv Abstracts (Clement et al., 2019), CC News (Charles et al., 2023; Hamborg et al., 2017), and Enron Emails

Dataset	**User Field**	**#Users**	**#Examples**	**Percentiles of Examples/User**				
				$\mathbf{P_0}$	$\mathbf{P_{25}}$	$\mathbf{P_{50}}$	$\mathbf{P_{75}}$	$\mathbf{P_{100}}$
ArXiv Abstracts	Submitter	16511	$625K$	20	24	30	41	3204
CC News	Domain Name	2839	$660K$	30	50	87	192	24480
Enron Emails	Email Address	150	$491K$	150	968	1632	3355	28229

Table 1: Evaluation dataset summary statistics: The three evaluation datasets vary in their notion of "user" (i.e. an ArXiv abstract belongs to the user who submitted it to ArXiv whereas a CC News article belongs to the web domain where the article was published). Additionally, these datasets span multiple orders of magnitude in terms of number of users and number of examples contributed per user.

(Klimt & Yang, 2004). Before fine-tuning models on these datasets we perform the following pre-processing steps to make them suitable for evaluating user inference.

1. We filter out users with fewer than a minimum number of samples (20, 30, and 150 samples for ArXiv, CC News, and Enron respectively). These thresholds were selected prior to any experiments to balance the following considerations: (1) each user must have enough data to provide the attacker with enough samples to make user inference feasible and (2) the filtering should not remove so many users that the fine-tuning dataset becomes too small. The summary statistics of each dataset after filtering are shown in Table 1.
2. We reserve 10% of the data for validation and test sets
3. We split the remaining 90% of samples into a held-in set and held-out set, each containing half of the users. The held-in set is used for fine-tuning models and the held-out set is used for attack evaluation.
4. For each user in the held-in and held-out sets, we reserve 10% of the samples as the attacker's knowledge about each user. These samples are never used for fine-tuning.

Target Models. We evaluate user inference attacks on the 125M and 1.3B parameter models from the GPT-Neo (Black et al., 2021) model suite. For each experiment, we fine-tune all parameters of these models for 10 epochs. We use the the Adam optimizer (Kingma & Ba, 2017) with a learning rate of $5\mathrm{e}{-5}$, a linearly decaying learning rate schedule with a warmup period of 200 steps, and a batch size of 8. After training, we select the checkpoint achieving the minimum loss on validation data from the users held in to training, and use this checkpoint to evaluate user inference attacks.

We train models on servers with one NVIDIA A100 GPU and 256 GB of memory. Each fine-tuning run took approximately 16 hours to complete for GPT-Neo 125M and 100 hours for GPT-Neo 1.3B.

Caveat to the experiments. Due to the size of The Pile, we found it challenging to find user-stratified datasets that were not part of model pre-training; this is a problem with LLMs in general (Sainz et al., 2023). However, we believe that our setup still faithfully evaluates the fine-tuning setting for two main reasons. First, the overlapping fine-tuning data constitutes only a small fraction of all the data in The Pile. Second, our attacks are likely only weakened (and thus, underestimate the true risk) by this setup. This is because inclusion of the held-out users in pre-training should only reduce the model's loss on these samples, making the loss difference smaller and thus our attack harder to employ.

Attack Evaluation. We evaluate attacks by computing the attack statistic from Section 2 for each held-in user that contributed data to the fine-tuning dataset, as well as the remaining held-out set of users. With these user-level statistics, we compute a Receiver Operating Characteristic (ROC) curve and report the area under this curve (AUROC) as our metric of attack performance. This metric has been used recently to evaluate the performance of membership inference attacks Carlini et al. (2022), and it provides a full spectrum of the attack effectiveness (True Positive Rates at fixed False Positive Rates). By reporting the AUROC, we do not need to select a threshold τ for our attack statistic, but rather we report the aggregate performance of the attack across all possible thresholds.

Canary User Construction. We evaluate worst-case risk of user inference by injecting synthetic canary users into the fine-tuning data from CC News and ArXiv Abstracts. These canaries were constructed by taking real users and replicating a shared substring in all of that user's examples.

This construction is meant to create canary users that are both realistic (i.e. not substantially outlying compared to the true user population) but also easy to perform user inference on. The algorithm used to construct canaries is shown in Algorithm 1.

Algorithm 1 Synthetic canary user construction

```
Input: Substring lengths $L = [l_1, \dots l_n]$, canaries per substring length $N$, set of real users $U_R$
Output: Set of canary users $U_C$
  $U_C \leftarrow \emptyset$
  for $l$ in $L$ do
      for $i$ up to $N$ do
          Uniformly sample user $u$ from $U_R$
          Uniformly sample example $x$ from $u$'s data
          Uniformly sample $l$-token substring $s$ from $x$
          $u_c \leftarrow \emptyset$                                ▷ Initialize canary user with no data
          for $x$ in $u$ do
              $x_c \leftarrow$ InsertSubstringAtRandomLocation$(x, s)$
              Add example $x_c$ to user $u_c$
          Add user $u_c$ to $U_C$
          Remove user $u$ from $U_R$
```

Mitigation Definitions. In Section 3.1 we explore heuristics for mitigating privacy attacks.

Batch gradient clipping restricts the norm of a single batch gradient to be at most C.

$$\hat{g}_t = \frac{\min(C, \|\nabla_{\theta_t} l(\boldsymbol{x})\|)}{\|\nabla_{\theta_t} l(\boldsymbol{x})\|} \nabla_{\theta_t} l(\boldsymbol{x})$$

Per-example gradient clipping restricts the norm of a single example's gradient to be at most C before aggregating the gradients into a batch gradient.

$$\hat{g}_t = \sum_{i=1}^{n} \frac{\min(C, \|\nabla_{\theta_t} l(\boldsymbol{x}^{(i)})\|)}{\|\nabla_{\theta_t} l(\boldsymbol{x}^{(i)})\|} \nabla_{\theta_t} l(\boldsymbol{x}^{(i)})$$

The batch or per-example clipped gradient $\hat{g}_t$, is then passed to the optimizer as if it were the true gradient.

For all experiments involving gradient clipping, we selected the clipping norm, C, by recording the gradient norms during a standard training run and setting C to the minimum gradient norm. In practice this resulted in clipping nearly all batch/per-example gradients during training.

D Additional Experimental Results

D.1 Attack Performance and Model Scale

We investigate the role of model scale in user inference. We fine-tune GPT-Neo 125M and 1.3B on CC News and evaluate attack performance.

We see from Figure 6, that the attack performance is nearly identical on both models with AUROCs of 65.3% for the 1.3B model and 65.8% for the 125M model. While the 1.3B parameter model achieves better validation loss on both held-in users (2.24 vs. 2.64) and held-out users (2.81 vs. 3.20), the generalization gap is nearly the same for both models (0.57 vs. 0.53). This shows a qualitative difference between user inference and membership inference, where in the latter threat model attack performance reliably increases with model size (Carlini et al., 2023; Kandpal et al., 2022; Mireshghallah et al., 2022; Tirumala et al., 2022).

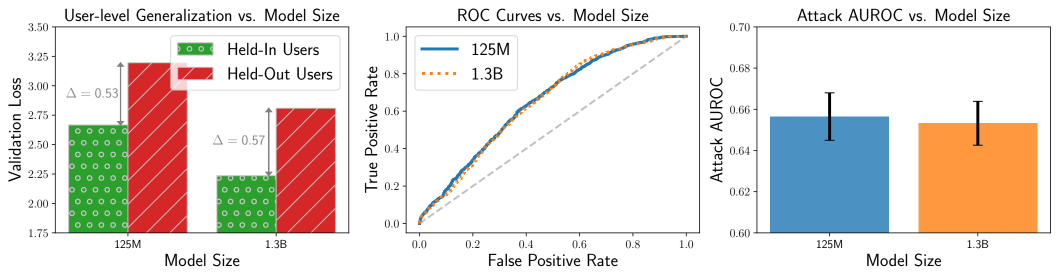

Figure 6: Attack performance as model size scales: User inference attack performance in 125M and 1.3B parameter models trained on CC News. **Left**: Although the 1.3B parameter model achieves lower validation loss, the difference in validation loss between held-in and held-out users is the same as that of the 125M parameter model. **Center and Right**: User inference attacks against the 125M and 1.3B parameter models achieve the same performance.

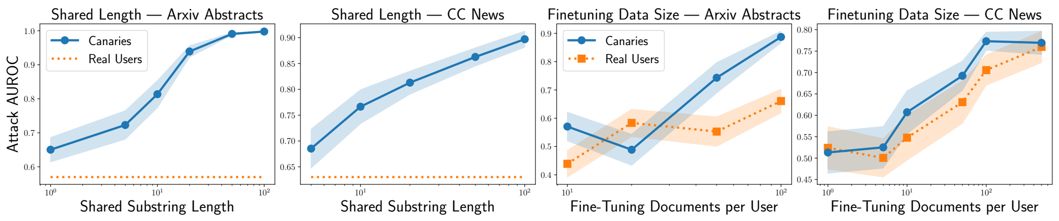

Figure 7: Canary experiments. **Left two**: Attack performance for canaries with different shared substring lengths. **Right two**: Attack performance on canary users and real users with different amounts of fine-tuning data per user. On all plots, we shade the AUROC standard deviation over 100 bootstrap samples of held-in and held-out users.

D.2 User Inference in the Worst-Case

The disproportionately large downside to privacy leakage necessitates looking beyond the average-case privacy risk to worst-case settings. To this end, we analyze attack performance on datasets containing synthetically generated users, known as *canaries*. There is usually a trade-off between making the canary users realistic and worsening their privacy risk. We intentionally err on the side of making them realistic to illustrate the potential risks of user inference.

To construct a canary user, we first sample a real user from the dataset and insert a particular substring into each of that user's examples. The substring shared between all of the user's examples is a contiguous substring randomly sampled from one of their documents (for more details, see Appendix C). We construct 180 canary users with shared substrings ranging from 1-100 tokens in length and inject these users into the ArXiv Abstracts and CC News datasets. We do not experiment with synthetic canaries in Enron Emails, as the attack AUROC already exceeds 92% for real users.

As expected, Figure 7 (left) shows that the attack effectiveness is significantly higher on canary users than real users, and increases monotonically with the length of the shared substring. However, we find that canaries with a short substring (5 tokens or smaller) is enough to significantly increase the attack AUROC from 57% to 72% for ArXiv and from 63% to 69% for CC News.

This increase of attack performance raises a question if canary gradients can be filtered out easily (e.g., using the ℓ_2 norm). However, Figure 5 (center-left) shows that the gradient norm distribution of the canary gradients and those of real users are nearly indistinguishable. This shows that our canaries are close to real users from the model's perspective, and thus hard to filter out. This experiment also demonstrates the increased privacy risk for users who use, for instance, a short and unique signature in emails or characteristic phrases in documents.

D.3 Ablations

We run additional ablations on the attack strategy and the reference model.

The user inference attacks implemented in the main paper use the pre-trained LLM as a reference model and compute the attack statistic as a mean of log-likelihood ratios described in Section 2. In this section, we study different choices of reference model and different methods of aggregating example-level log-likelihood ratios. For each of the attack evaluation datasets, we test different choices of reference model and aggregation function for performing user inference on a fine-tuned GPT-Neo 125M model.

In Table 2 we test three methods of aggregating example-level statistics and find that averaging taking the average log-likelihood ratio outperforms using the minimum or maximum example. Additionally, in Table 3 we find that using the pre-trained GPT-Neo model as the reference model outperforms using an independently trained model of equivalent size. However, in the case that an attacker does not know or have access to the pre-trained model, using an independently trained LLM as a reference still yields strong attack performance.

Attack Statistic Aggregation	**ArXiv Abstracts**	**CC News**	**Enron Emails**
Mean	$\mathbf{57.2 \pm 0.4}$	$\mathbf{65.7 \pm 1.1}$	$\mathbf{92.7 \pm 2.0}$
Max	$\mathbf{56.7 \pm 0.4}$	62.1 ± 1.1	79.7 ± 3.3
Min	55.3 ± 0.4	63.3 ± 1.0	86.8 ± 2.9

Table 2: Attack statistic design: We compare the default mean aggregation of per-document statistics $\log(p_\theta(\boldsymbol{x}^{(i)})/p_{\mathsf{ref}}(\boldsymbol{x}^{(i)}))$ in the attack statistic (§2) with the min/max over documents $i = 1, \dots, m$. We show the mean and std AUROC over 100 bootstrap samples of the held-in and held-out users.

Reference Model	**ArXiv Abstracts**	**CC News**	**Enron Emails**
GPT-Neo 125M*	$\mathbf{57.2 \pm 0.4}$	$\mathbf{65.8 \pm 1.1}$	$\mathbf{93.1 \pm 1.9}$
GPT-2 124M	53.1 ± 0.5	$\mathbf{65.7 \pm 1.2}$	87.2 ± 2.7
OPT 125M	53.7 ± 0.5	62.0 ± 1.2	87.6 ± 3.2

Table 3: Effect of the reference model: We show the user inference attack AUROC (%) for different choices of the reference model p_{ref}, including the pretrained model p_{θ_0} (GPT-Neo 125M, denoted by *). We show the mean and std AUROC over 100 bootstrap samples of the held-in and held-out users.