User Inference Attacks on Large Language Models

Nikhil Kandpal[1] **Krishna Pillutla**[2] **Alina Oprea**[3]
Peter Kairouz[4] **Christopher A. Choquette-Choo**[4] **Zheng Xu**[4]

[1]University of Toronto & Vector Institute [2]Indian Institute of Technology (IIT) Madras
[3]Northeastern University [4]Google

Abstract

Text written by humans makes up the vast majority of the data used to pre-train and fine-tune large language models (LLMs). Many sources of this data—like code, forum posts, personal websites, and books—are easily attributed to one or a few "users". In this paper, we ask if it is possible to infer if any of a *user's* data was used to train an LLM. Not only would this constitute a breach of privacy, but it would also enable users to detect when their data was used for training. We develop the first effective attacks for *user inference*—at times, with near-perfect success—against LLMs. Our attacks are easy to employ, requiring only black-box access to an LLM and a few samples from the user, which *need not be the ones that were trained on*. We find, both theoretically and empirically, that certain properties make users more susceptible to user inference: being an outlier, having highly correlated examples, and contributing a larger fraction of data. Based on these findings, we identify several methods for mitigating user inference including training with example-level differential privacy, removing within-user duplicate examples, and reducing a user's contribution to the training data. Though these provide partial mitigation, our work highlights the need to develop methods to fully protect LLMs from user inference.

1 Introduction

LLMs like GPT-4 (OpenAI, 2023) and Gemini (Team et al., 2023) have achieved phenomenal success. Yet, the datasets used to train LLMs are often comprised of user-written data. This raises an important question: can we detect (or, infer) if a user's data was used to train a model? If so, this would enable data-owners to detect usage of their data. More concerningly, this would also constitute a breach of user privacy. In this paper, we investigate the latter privacy concern.

To do this, we focus on the setting of fine-tuning on domain-specific data, one of the most widely used methods for applying LLMs to real-world problems (Liu et al., 2022; Mosbach et al., 2023), with several commerical products deployed today using this approach, e.g., GitHub Copilot (Chen et al., 2021), Gmail Smart Compose (Chen et al., 2019), and GBoard (Xu et al., 2023). In this setting, it is not uncommon for fine-tuning data to include potentially sensitive user data.

In this work, we show that user inference (Figure 1) is a realistic privacy attack for LLMs fine-tuned on user data by constructing a simple and practical attack to determine if a user participated in fine-tuning. Our attack involves computing a likelihood ratio test statistic normalized relative to a reference model (§3), which can be performed efficiently even at the LLM scale. We empirically study its effectiveness on the GPT-Neo family of LLMs (Black et al., 2021) when fine-tuned on diverse data domains, including emails, social media comments, and news articles (§4.2). This study gives insight into the various parameters that affect vulnerability to user inference—such as uniqueness of a user's data distribution, amount of fine-tuning data contributed by a user, and amount of attacker knowledge about a user.

Notably, our attack requires only a few *fresh* samples from a user, i.e., not necessarily ones used in model training. This significantly improves on the assumptions of other attacks, like membership inference, in which the attacker is assumed to already have a set of samples that includes some training data (Mireshghallah et al., 2022; Mattern et al., 2023; Niu et al., 2023). Beyond this, our attacks also allow us to estimate the privacy leakage of a collection of samples written by a single user that may share characteristics (e.g., writing style, topic, etc.). This can not be quantified by LLM privacy attacks like membership inference or extraction attacks (Carlini et al., 2021; Lukas et al.,

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 18238–18265
November 12-16, 2024

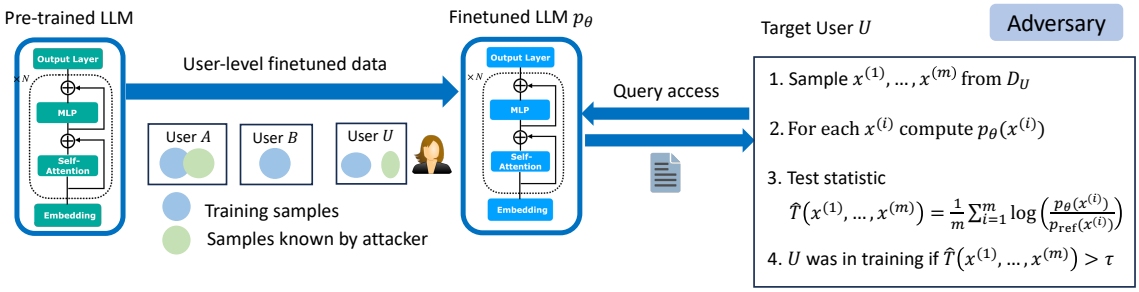


Figure 1: The user inference threat model. An LLM is fine-tuned on user-stratified data. The adversary can query the fine-tuned model to compute likelihoods. The adversary can access samples from a user's distribution (different than the user training samples) to compute a likelihood score to determine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

2023; Carlini et al., 2023), as they make no assumptions about the data's origin.

Additionally, we evaluate the attack on synthetically generated canary users to characterize the privacy leakage for worst-case users (§4.3). We show that canary users constructed via minimal modifications to the real users' data increase the attack's effectiveness (in AUROC) by up to 40%. This construction indicates that simple features shared across a user's samples like an email signature or a characteristic phrase, can greatly exacerbate the risk of user inference.

Finally, we evaluate strategies for mitigating user inference, such as limiting the number of samples contributed by each user, removing duplicates within a user's samples, early stopping, gradient clipping, and example-level differential privacy (DP). Our results show that duplicates within a user's examples can exacerbate the risk of user inference, but are not necessary for a successful attack. Limiting a user's contribution to the fine-tuning set can be effective but is only feasible for data-rich applications with a large number of users. Finally, example-level DP provides some defense but is ultimately designed to protect the privacy of individual examples, rather than users that contribute multiple examples. These results highlight the importance of future work on scalable *user-level* DP algorithms that can provably mitigate user inference (McMahan et al., 2018; Asher et al., 2021; Charles et al., 2024). Overall, we are the first to study user inference against LLMs and provide key insights to inform future deployments of LLMs fine-tuned on user data.

2 Related Work

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

Membership inference attacks on LLMs. 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 prior works. 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 sequences copied from the training set. Note that membership inference requires access to exact training samples while user inference does not.

Extraction attacks. Memorization in LLMs received much attention (Carlini et al., 2021; Zhang et al., 2023; Tirumala et al., 2022; Biderman et al., 2023; Ippolito et al., 2023; Anil et al., 2023). These works found that memorization scales with model size (Carlini et al., 2023) and data repetition (Kandpal et al., 2022), may eventually be forgotten (Jagielski et al., 2023), and can exist even on

models trained for restricted use-cases like translation (Kudugunta et al., 2023). Lukas et al. (2023) develop techniques to extract PII information from LLMs and (Inan et al., 2021) design metrics to measure the leakage of user's confidential data by the LLM. Once a user's participation is identified by user inference, these techniques can be used to estimate the amount of privacy leakage.

User-level membership inference. Much work on inferring a user's participation in training makes the stronger assumption that the attacker has access to the user's exact training samples. We call this *user-level membership inference* (to contrast with *user inference* which does not require the exact training samples). Song and Shmatikov (2019) gave the first attack of this kind for generative text models. However, their attack trains multiple shadow models and does not scale to LLMs. Shejwalkar et al. (2021) study this threat model for text classification via reduction to membership inference.

User inference. This threat model was considered for speech recognition (Miao et al., 2021), representation learning (Li et al., 2022) and face recognition (Chen et al., 2023). Hartmann et al. (2023) formally define user inference for classification and regression but call it *distributional membership inference*. These attacks are domain-specific or require shadow models, and do not apply or scale to LLMs. Instead, we design an efficient user inference attack that scales to LLMs and illustrate the user-level privacy risks posed by fine-tuning on user data. See Appendix C for further discussion of other related threat models such as property inference and authorship attribution.

3 User Inference Attacks

An autoregressive language model p_θ defines a distribution $p_\theta(x_t|\boldsymbol{x}_{<t})$ over the next token x_t in continuation of a prefix $\boldsymbol{x}_{<t} \doteq (x_1, \ldots, x_{t-1})$. We focus on fine-tuning, where a pre-trained LLM p_{θ_0} (with initial parameters θ_0) is trained on a dataset D_{FT} sampled i.i.d. from a distribution $\mathcal{D}_{\mathsf{task}}$. The canonical objective is to minimize the cross entropy of predicting each next token x_t given the context $\boldsymbol{x}_{<t}$ for each fine-tuning sample $\boldsymbol{x} \in D_{\mathsf{FT}}$. Thus, the fine-tuned model p_θ is trained to maximize the log-likelihood $\sum_{\boldsymbol{x}\in D_{\mathsf{FT}}} \log p_\theta(\boldsymbol{x}) = \sum_{\boldsymbol{x}\in D_{\mathsf{FT}}} \sum_{t=1}^{|\boldsymbol{x}|} \log p_\theta(x_t|\boldsymbol{x}_{<t})$ of the dataset D_{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 their author. 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 user stratification, we model the fine-tuning distribution $\mathcal{D}_{\mathsf{task}}$ as a mixture

$$\mathcal{D}_{\mathsf{task}} = \sum_{u=1}^{n} \alpha_u \mathcal{D}_u \tag{1}$$

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. One can sample from $\mathcal{D}_{\mathsf{task}}$ by first sampling a user u with probability α_u and then sampling a document $\boldsymbol{x} \sim \mathcal{D}_u$ from the user's data distribution. 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 access to a text sample $\boldsymbol{x}$ and must determine whether $\boldsymbol{x}$ was a part of the fine-tuning data (Shokri et al., 2017; Yeom et al., 2018; Carlini et al., 2022). The **user inference** threat model relaxes the stringent assumption that the attacker has access to samples from the fine-tuning data.

The attacker aims to determine if *any* data from user u was involved in fine-tuning the model p_θ using m i.i.d. samples $\boldsymbol{x}^{(1:m)} := (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) \sim \mathcal{D}_u^m$ from user u's distribution. Crucially, we allow $\boldsymbol{x}^{(i)} \notin D_{\mathsf{FT}}$, i.e., the attacker is not assumed to have access to the exact samples of user u that were a part of the fine-tuning set. 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. This is a realistic threat model for LLMs as it does not require *exact* knowledge of a user's training set samples, as in membership inference attacks.

We adopt the black-box threat model (Salem et al., 2023; Jia et al., 2019) and assume that the attacker can only query the model's likelihood on a sequence of tokens, but may not know either the model architecture or parameters.[1] Following standard practice in membership infer-

[1]This differs from the API-only threat model in that we require the model's likelihoods; our use of the term *black-box* is consistent with the membership inference literature.

ence (Mireshghallah et al., 2022; Watson et al., 2022), we allow the attacker access to a reference model p_{ref} that is similar to the target model p_θ but has not been fine-tuned on user u's data. This can be the 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 aims to test:

$$H_0 : p_\theta \notin \mathcal{P}_u, \qquad H_1 : p_\theta \in \mathcal{P}_u\,. \tag{2}$$

There is generally no prescribed recipe to test for such a composite hypothesis. Typical attack strategies involve training multiple "shadow" models (Shokri et al., 2017); see §B. This, however, is infeasible at LLM scale.

The likelihood under the fine-tuned model p_θ is a natural test statistic: we might expect $p_\theta(\boldsymbol{x}^{(i)})$ to be high if H_1 is true and low otherwise. Unfortunately, this is not always true, even for membership inference. Indeed, $p_\theta(\boldsymbol{x})$ can be large for $\boldsymbol{x} \notin D_{\mathsf{FT}}$ for easy-to-predict text sequence $\boldsymbol{x}$ (e.g., generic text using common words), while $p_\theta(\boldsymbol{x})$ can be small even if $\boldsymbol{x} \in D_{\mathsf{FT}}$ for hard-to-predict $\boldsymbol{x}$. This necessitates calibrating the test using a reference model (Mireshghallah et al., 2022; Watson et al., 2022).

We overcome this difficulty by replacing the attacker's task with surrogate hypotheses that are easier to test efficiently:

$$H_0' : \boldsymbol{x}^{(1:m)} \sim p_{\mathsf{ref}}^m\,, \quad H_1' : \boldsymbol{x}^{(1:m)} \sim p_\theta^m\,. \tag{3}$$

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 fine-tuning dataset D_{FT} are similar to the samples $\boldsymbol{x}^{(1:m)}$ known to the attacker even if they are not identical. In this case, the attacker rejects H_0'. Conversely, if user u did not participate in fine-tuning and no samples from D_{FT} are similar to $\boldsymbol{x}^{(1:m)}$, 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 that $\boldsymbol{x}, \boldsymbol{x}' \sim \mathcal{D}_u$ are closer on average than $\boldsymbol{x} \sim \mathcal{D}_u$ and $\boldsymbol{x}'' \sim \mathcal{D}_{u'}$ for any $u' \neq u$.

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 (Lehmann et al., 1986, Thm. 3.2.1). This involves constructing a test statistic using the log-likelihood ratio

$$\begin{aligned} T\big(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)}\big) &:= \log\left(\frac{p_\theta(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)})}{p_{\mathsf{ref}}(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)})}\right) \\ &= \sum_{i=1}^{m} \log\left(\frac{p_\theta(\boldsymbol{x}^{(i)})}{p_{\mathsf{ref}}(\boldsymbol{x}^{(i)})}\right)\,, \end{aligned} \tag{4}$$

where the last equality follows from the independence of each $\boldsymbol{x}^{(i)}$, which we assume. Although independence may be violated in some domains (e.g. email threads), it makes the problem more computationally tractable. As we shall see, this already gives us relatively strong attacks.

Given a threshold τ, the attacker rejects the null hypothesis and declares that u has participated in fine-tuning if $T(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)}) > \tau$. 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 $\hat{T}(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)}) = \frac{1}{m}\,T(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)})$.

For $m = 1$, our test statistic reduces to $\hat{T}(\boldsymbol{x}) = \log(p_\theta(\boldsymbol{x})/p_{\mathsf{ref}}(\boldsymbol{x}))$, which is a common test statistic for membership inference of the sample $\boldsymbol{x}$ (Carlini et al., 2021). Thus, our test statistic $\hat{T}(\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)})$ can be interpreted as averaging the membership inference statistic over the m samples $\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(m)}$.

Analysis. We analyze this attack statistic in a simplified setting to gain some intuition. In the large sample limit as $m \to \infty$, the mean statistic $\hat{T}$ approximates the population average

$$\bar{T}(\mathcal{D}_u) := \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}_u}\left[\log\left(\frac{p_\theta(\boldsymbol{x})}{p_{\mathsf{ref}}(\boldsymbol{x})}\right)\right]\,. \tag{5}$$

We will analyze this test statistic for the choice $p_{\mathsf{ref}} = \mathcal{D}_{-u} \propto \sum_{u'\neq u} \alpha_{u'}\mathcal{D}_{u'}$, 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 χ^2 divergences. We establish a bound (proved in §A) assuming $p_\theta, p_{\mathsf{ref}}$ 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*

$$\begin{aligned} &\bar{T}(\mathcal{D}_u) \leq \alpha_u\,\chi^2(\mathcal{D}_u\|\mathcal{D}_{-u})\,, \quad \textit{and} \\ &\bar{T}(\mathcal{D}_u) > \log(\alpha_u) + \mathrm{KL}(\mathcal{D}_u \,\|\, \mathcal{D}_{-u})\,. \end{aligned}$$

Dataset	User Field	#Users	#Examples	Percentiles of Examples/User				
				$\mathbf{P_0}$	$\mathbf{P_{25}}$	$\mathbf{P_{50}}$	$\mathbf{P_{75}}$	$\mathbf{P_{100}}$
Reddit Comments	User Name	5194	$1002K$	100	116	144	199	1921
CC News	Domain Name	2839	$660K$	30	50	87	192	24480
Enron Emails	Sender's Email Address	136	$91K$	28	107	279	604	4280

Table 1: Evaluation dataset summary statistics: The three evaluation datasets vary in their notion of "user" (a Reddit comment belongs to the poster's username whereas a CC News article belongs to the its web domain). Additionally, these datasets span multiple orders of magnitude in terms of number of users and number of examples contributed per user.

This suggests the attacker may more easily infer:
(a) users who contribute more data (large α_u), or
(b) users who contribute unique data (so $\mathrm{KL}(\mathcal{D}_u \| \mathcal{D}_{-u})$ and $\chi^2(\mathcal{D}_u \| \mathcal{D}_{-u})$ are large).
Conversely, if neither holds, then a user's participation in fine-tuning cannot be reliably detected. Our experiments corroborate these and we use them to design potential mitigation strategies.

4 Experiments

In this section, we empirically study the susceptibility of models to user inference attacks, the factors that affect attack performance, and potential mitigation strategies. For this, we fine-tune LLMs on our user data as this enables us to rigorously test the leakage and because fine-tuning is a common strategy for adapting to user data(see Section 1). We believe our results would extend even to models pre-trained on user data but leave such an evaluation to future work.

4.1 Experimental Setup

Datasets. We evaluate user inference on 3 user-stratified text datasets: Reddit Comments (Baumgartner et al., 2020) for social media content, CC News[2] (Hamborg et al., 2017) for news articles, and Enron Emails (Klimt and Yang, 2004) for user emails. These datasets are diverse in their domain, notion of a user, number of users, and data per user (Table 1). We also report results for the ArXiv Abstracts dataset (Clement et al., 2019) in §E.

To make these datasets suitable for evaluating user inference, we split them into a held-in set of users to fine-tune models, and a held-out set of users to evaluate attacks. Additionally, we set aside 10% of each user's samples as the samples used by the attacker to run user inference attacks; these samples are not used for fine-tuning. For more details on the dataset preprocessing, see §D.

Models. We evaluate user inference attacks on the 125M and 1.3B parameter decoder-only LLMs from the GPT-Neo (Black et al., 2021) model suite. These models were pre-trained on The Pile dataset (Gao et al., 2020), an 825 GB diverse text corpus, and use the same architecture and pre-training objectives as the GPT-2/GPT-3 models. Further details on the fine-tuning are given in §D.

Attack and Evaluation. We implement the user inference attack of Section 3 using the pre-trained GPT-Neo models as the reference p_{ref}. Following the membership inference literature, 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 rate (TPR) and false positive rate (FPR) of the attack across all possible thresholds. We use the area under this curve (AUROC) as a scalar summary. We also report the TPR at small FPR (e.g., 1%) (Carlini et al., 2022).

4.2 User Inference: Results and Properties

We 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.

Attack Performance. We attack GPT-Neo 125M fine-tuned on each of the three fine-tuning datasets and evaluate 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 88% (Enron) to 66% (CC News) and 56% (Reddit).

The disparity in performance between the three datasets can be explained in part by the intuition from Proposition 1, which points out two factors. First, a larger fraction of data contributed by a user makes user inference easier. The Enron dataset has fewer users, each of whom contributes a significant fraction of the fine-tuning data (cf. Table 1),

[2]While CC News does not strictly have user data, it is made up of non-identical groups based on the web domain. We treat each group as a "user" as in Charles et al. (2023).

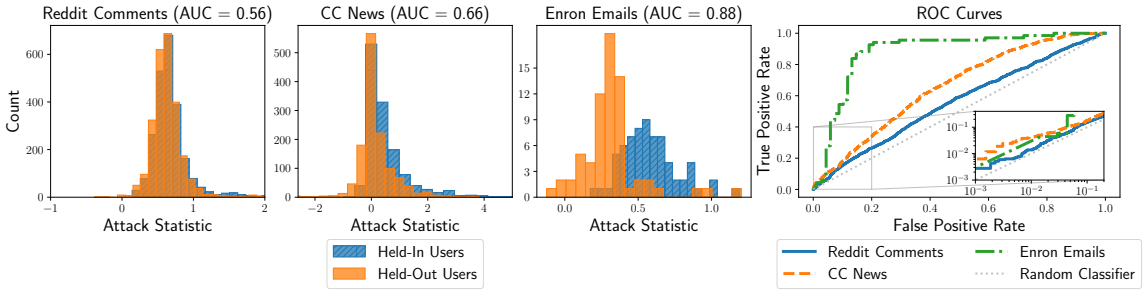


Figure 2: Our attack can achieve significant AUROC, e.g., on the Enron emails dataset. **Left three**: Histograms of the test statistics for held-in and held-out users for the three attack evaluation datasets. **Rightmost**: Their corresponding ROC curves.

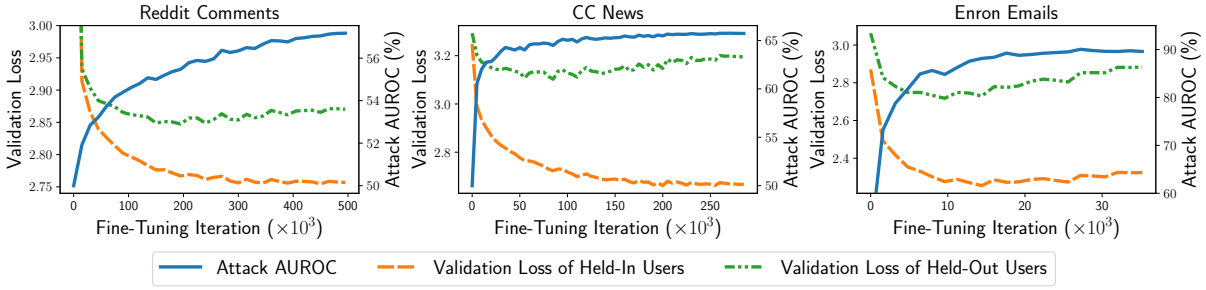


Figure 3: Attack success over fine-tuning: User inference AUROC and the held-in/held-out validation loss.

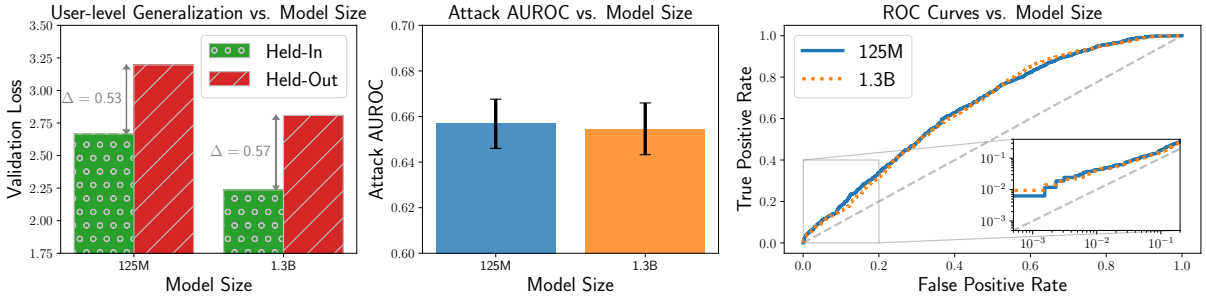


Figure 4: Attack success vs. model scale: User inference attack performance in 125M and 1.3B models trained on CC News. **Left**: Although the 1.3B model achieves lower validation loss, the difference in validation loss between held-in and held-out users is the same as that of the 125M model. **Center & Right**: User inference attacks against the 125M and 1.3B models achieve the same performance.

while, the Reddit dataset has a large number of users, each with few datapoints. Second, distinct user data makes user inference easier. Emails are more distinct due to identifying information such as names (in salutations and signatures) and addresses, while news articles or social media comments from a particular user may share more subtle features like topic or writing style.

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 3 that the validation loss of held-in users continues to decrease for *all 3 datasets*, 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. Note 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 datasets. This demonstrates the close relation between user-level overfitting and user inference.

Attack Performance and Model Scale. Next, we

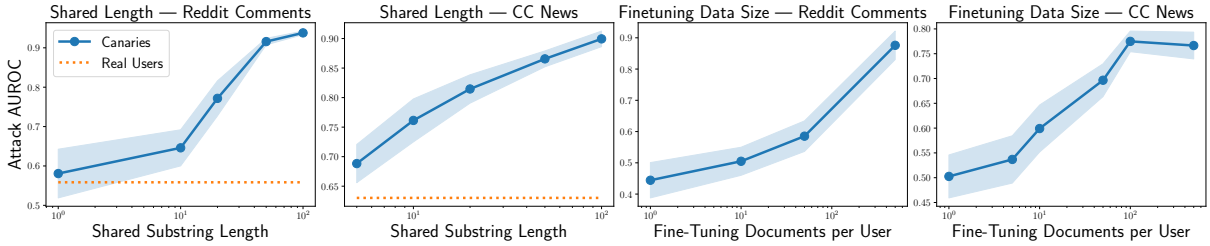


Figure 5: Canary experiments. **Left two**: Comparison of attack performance on the natural distribution of users ("Real Users") and attack performance on synthetic canary users (each with 100 fine-tuning documents) as the shared substring in a canary's documents varies in length. **Right two**: Attack performance on canary users (each with a 10-token shared substring) decreases as their contribution to the fine-tuning set decreases. The shaded region denotes std over 100 bootstrap samples.

investigate the role of model scale in user inference using the GPT-Neo 125M and 1.3B on the CC News dataset. We find in Figure 4 that the attack AUROC is nearly identical for the 1.3B model (65.3%) and 125M model (65.8%). While the larger 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 highlights a qualitative difference between user and membership inference, where attack performance reliably increases with model size in the latter (Carlini et al., 2023; Tirumala et al., 2022; Kandpal et al., 2022; Mireshghallah et al., 2022).

Effect of the Attacker Knowledge. We examine the effect of the attacker knowledge (the amount of user data used by the attacker to compute the test statistic) in Figure 10 of §E. First, we find that more attacker knowledge leads to higher attack AUROC and lower variance in the attack success. For CC News, the AUROC increases from $62.0 \pm 3.3\%$ when the attacker has only one document to $68.1 \pm 0.6\%$ at 50 documents. 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%). Overall, the results show that an attacker does not need much data to mount a strong attack, and more data only helps.

4.3 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. Thus, we analyze attack performance on datasets containing synthetically generated users, known as *canaries*. There is a trade-off between making the canary users realistic and worsening their privacy risk. We 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 §D). We construct 180 canary users with shared substrings ranging from 1-100 tokens in length and inject these users into the Reddit and CC News datasets. We do not experiment with synthetic canaries in Enron Emails, as the attack AUROC already exceeds 88% for real users.

Figure 5 (left) shows that the attack is more effective on canaries than real users, and increases with the length of the shared substring. A short shared substring is enough to increase the attack AUROC from 63% to 69% (5 tokens) for CC News and 56% to 65% for Reddit (10 tokens).

4.4 Mitigation Strategies

We investigate several heursitics for limiting the influence of individual examples or users on fine-tuning as methods for mitigating user inference.

Early Stopping. The connection between user inference and user-level overfitting from §4.2 suggests that early stopping, a common heuristic used to prevent overfitting (Caruana et al., 2000), could potentially mitigate 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 (18% of the fine-tuning) for Reddit, see Figure 3. Typically, the overall validation loss still decreases far after this point. This suggests an explicit tradeoff between model utility (e.g., in validation loss) and privacy risks from user inference.

Data Limits Per User. To mitigate user inference, we consider limiting the amount of fine-tuning

data per user. Figure 5 (right two) show that this can be effective. For CC News, the AUROC for canary users reduces from 77% at 100 documents per user to almost random chance at 5 documents per user. A similar trend also holds for Reddit.

Data Deduplication. Since data deduplication can mitigate membership inference (Lee et al., 2022; Kandpal et al., 2022), we evaluate it for user inference. CC News is the only dataset in our suite with within-user duplicates (Reddit and Enron are deduplicated in the preprocessing; see Appendix D.1), so we use it for this experiment.[3] The deduplication reduces the attack AUROC from 65.7% to 59.1%. The attack ROC curve of the deduplicated version is also uniformly lower, even at extremely small FPRs (Figure 13 of §E).

Thus, data *repetition* can exacerbate user inference. However, results on Reddit and Enron Emails (no duplicates) suggest that deduplication is insufficient to fully mitigate user inference.

Example-level Differential Privacy (DP). DP (Dwork et al., 2006) gives provable bounds on privacy leakage. We study how example-level DP, which protects the privacy of individual *examples*, impacts *user* inference. We train the 125M model on Enron Emails using DP-Adam, a variant of Adam that clips per-example gradients and adds noise calibrated to the privacy budget ε. We find next that example-level DP can somewhat mitigate user inference while incurring increased compute cost and a degraded model utility.

Obtaining good utility with DP requires large batches and more epochs (Ponomareva et al., 2023), so we use a batch size of 1024, tune the learning rate, and train the model for 50 epochs ($1.2K$ updates), so that each job runs in 24h (in comparison, non-private training takes 1.5h for 7 epochs); details of the tuning are given in §D.4.

Table 2 shows a severe degradation in the validation loss under DP. For instance, a loss of 2.67 at the weak guarantee of $\varepsilon = 32$ is surpassed after just $1/3^{\text{rd}}$ of an epoch of non-private training; this loss continues to reduce to 2.43 after 3 epochs. In terms of attack effectiveness, example-level DP reduces the attack AUROC and the TPR at FPR $= 5\%$, while the TPR at FPR $= 1\%$ remains the same or gets worse. Indeed, while example-level DP protects individual examples, it can fail to protect the privacy of *users* who contribute many examples. This highlights the need for scalable algorithms and software for fine-tuning LLMs with *DP at the user-level*. Currently, user-level DP algorithms have been designed for small models in federated learning, but do not yet scale to LLMs.

Metric	$\varepsilon = 2$	$\varepsilon = 8$	$\varepsilon = 32$	Non-private
Val. Loss	2.77	2.71	2.67	2.43
Attack AUROC	64.7%	66.7%	67.9%	88.1%
TPR @ FPR= 1%	8.8%	8.8%	10.3%	4.4%
TPR @ FPR= 5%	11.8%	10.3%	10.3%	27.9%

Table 2: Example-level differential privacy: Training a model on Enron Emails under $(\varepsilon, 10^{-6})$-DP at the example-level (smaller ε implies a higher level of privacy).

Gradient Clipping. To avoid the degradation in performance with DP, we ask if it suffices to clip the gradients (without adding noise as in DP), at the example-level or the batch-level (Pascanu et al., 2013). The results are given in Figure 12 of §E: the left plot shows that neither batch nor per-example gradient clipping affect user inference. The right plot tells us why: canary examples do *not* have large outlying gradients and clipping affects real and canary data similarly. Thus, gradient clipping is an ineffective mitigation strategy.

Summary. Our results show that user inference is hard to mitigate with common heuristics. Careful deduplication is necessary to ensure that data repetition does not exacerbate user inference. Enforcing data limits per user can be inexpensive to implement and effective to mitigate user inference but only works for applications with a large number of users. Example-level DP can offer some mitigation at the cost of degraded model utility and greatly increased computation cost. Developing a feasible and effective mitigation strategy that also works efficiently in data-scarce applications remains an open problem.

5 Discussion, Limitations, Future Work

Fine-tuning LLMs on user data is a natural choice because these data typically resemble the types of inputs an LLM will encounter in deployment. We show this also exposes new risks for privacy leakage, making it easy to infer if a user's data was used for fine-tuning. Our proposed user inference attack achieves this by aggregating statistics across a user's data that leverage correlations between texts. We find that this attack can reliably

[3] Although each article of CC News from HuggingFace Datasets has a unique URL, the text of 11% of the articles has exact duplicates from the same domain. See §D.5 for examples.

detect a user's presence in the fine-tuning data, *even without access to their contributed training data*. Our work underscores the need for scaling user-aware training pipelines, such as user-level DP, to handle large datasets and models. We now discuss the limitations of our work and point out promising avenues for future research.

Overlap in Pre-Training and Fine-Tuning Data. The threat model studied in this paper aims to compromise user privacy when an LLM is fine-tuned on user-stratified data. However, due to the rapidly increasing size of commonly used pre-training datasets, the fine-tuning domains studied in this work, and often the specific fine-tuning datasets, are also present in LLMs' pre-training data. In particular, each of the fine-tuning datasets used in our experiments is also present to some extent in The Pile dataset (Gao et al., 2020) used to pre-train the GPT-Neo family of models (Black et al., 2021). Thus, one limitation of this work is that it only evaluates the user inference attack on fine-tuning datasets that, at least partially, overlap with the target LLM's pre-training data.

Despite this limitation, we believe that our setup still faithfully evaluates the effectiveness of user inference attacks. First, the overlapping fine-tuning data constitutes only a tiny fraction of the pre-training dataset. Second, our attacks are likely weakened (and thus, underestimate the true risk) in this setup because data from both held-in and held-out users are seen during pre-training. The inclusion of held-out users' data in pre-training should only reduce the model's loss on these samples, making the difference in loss after fine-tuning between held-in and held-out users smaller.

Furthermore, evaluating using fine-tuning datasets that overlap with pre-training data may actually be realistic for the way that some LLMs are trained today. Past work has shown that dataset contamination, where downstream evaluation datasets are found to be present in pre-training datasets, plagues many modern LLM pre-training datasets (Sainz et al., 2023; Oren et al., 2024; Dodge et al., 2021). Thus, unbeknownst to practitioners, fine-tuning datasets could be present in pre-training datasets, mirroring the evaluation setup in this paper. Recent work on pre-training has also shown that intentionally including typical fine-tuning data in the late stages of pre-training is beneficial (Hu et al., 2024).

Fine-tuning versus Pre-Training. Our results focus exclusively on models fine-tuned with user data. While this is a common setup deployed even in production models (cf. §1), pre-training datasets can also include user data. We leave a rigorous evaluation in this setup to future work, and note that this setting may be more difficult than the fine-tuning setting (Duan et al., 2024).

Other Threat Models. Our black-box threat model assumes that the likelihood (or equivalently, its loss) of a sequence under the model can be queried. However, some LLMs are only accessible via an API; it is interesting to consider user inference attacks in this setting. Some APIs expose the model's logits (or log-probabilities) from which likelihoods can be reconstructed (Finlayson et al., 2024); this allows for a variant of our user inference attack to be mounted.

Designing user inference attacks for APIs that do not expose model likelihoods/losses is an open question. In fact, defining membership inference attacks in this setting, or defining the equivalent of label-only membership inference attacks (Choquette-Choo et al., 2021) for LLMs are also open questions. Our work provides a recipe to lift future progress on such membership inference attacks to user inference: if $T(\boldsymbol{x})$ is a membership inference test statistic for a sequence $\boldsymbol{x}$, then $\hat{T}(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) = (1/m) \sum_{i=1}^{m} T(\boldsymbol{x}^{(i)})$ can be used a user inference test statistic; see §3. We leave this as a promising direction for future work.

Mitigating Attacks with User-Level DP. User-level DP is the gold standard defense against privacy attacks that aim to expose user participation in training. Implementing user-level DP as a defense in our experimental setup presents many challenges, ranging from fundamental dataset size limitations, software/systems challenges to allow user-level DP training to scale to LLMs, and a lack of understanding of the empirical tradeoffs needed to train performant user-level DP models. For a more thorough discussion of user-level DP and the challenges preventing its use in this setting, see Appendix F. The follow-up works of Charles et al. (2024) and Chua et al. (2024) show some promising progress on this front.

Finally, leveraging user inference attacks to audit user-level DP, like membership inference attacks for example-level DP (Jagielski et al., 2020; Pillutla et al., 2023; Steinke et al., 2023), is a promising future direction.

References

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

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

Giuseppe Ateniese, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, Domenico Vitali, and Giovanni Felici. 2015. Hacking Smart Machines with Smarter Ones: How to Extract Meaningful Data from Machine Learning Classifiers. *Int. J. Secur. Networks*, 10(3):137–150.

Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. *Proceedings of the International AAAI Conference on Web and Social Media*, 14(1):830–839.

Stella Biderman, USVSN Sai Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raf. 2023. Emergent and Predictable Memorization in Large Language Models. *arXiv:2304.11158*.

Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. 2022. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy*.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *ICLR*.

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. 2021. Extracting training data from large language models. In *USENIX*.

Rich Caruana, Steve Lawrence, and C Giles. 2000. Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping . *NeurIPS*.

Zachary Charles, Arun Ganesh, Ryan McKenna, H Brendan McMahan, Nicole Mitchell, Krishna Pillutla, and Keith Rush. 2024. Fine-Tuning Large Language Models with User-Level Differential Privacy. *arXiv Preprint*.

Zachary Charles, Nicole Mitchell, Krishna Pillutla, Michael Reneer, and Zachary Garrett. 2023. Towards Federated Foundation Models: Scalable Dataset Pipelines for Group-Structured Learning. In *NeurIPS (Datasets and Benchmarks Track)*.

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. 2021. Evaluating Large Language Models Trained on Code. *arXiv Preprint*.

Mia Xu Chen, Benjamin N. Lee, Gagan Bansal, Yuan Cao, Shuyuan Zhang, Justin Lu, Jackie Tsay, Yinan Wang, Andrew M. Dai, Zhifeng Chen, Timothy Sohn, and Yonghui Wu. 2019. Gmail Smart Compose: Real-Time Assisted Writing. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, and Yang Zhang. 2023. FACE-AUDITOR: Data auditing in facial recognition systems. In *USENIX Security*, pages 7195–7212, Anaheim, CA. USENIX Association.

Christopher A Choquette-Choo, Krishnamurthy Dvijotham, Krishna Pillutla, Arun Ganesh, Thomas Steinke, and Abhradeep Thakurta. 2024. Correlated Noise Provably Beats Independent Noise for Differentially Private Learning. In *ICLR*.

Christopher A Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. 2021. Label-only membership inference attacks. In *ICML*.

Lynn Chua, Badih Ghazi, Yangsibo Huang, Pritish Kamath, Daogao Liu, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. 2024. Mind the Privacy Unit! User-Level Differential Privacy for Language Model Fine-Tuning. *arXiv Preprint*.

Colin B. Clement, Matthew Bierbaum, Kevin P. O'Keeffe, and Alexander A. Alemi. 2019. On the Use of ArXiv as a Dataset. *arXiv Preprint*.

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

Jesse Dodge, Maarten Sap, Ana Marasovic, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus. In *EMNLP*, pages 1286–1305.

Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. 2024. Do Membership Inference Attacks Work on Large Language Models? In *COLM*.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Proc. of the Third Conf. on Theory of Cryptography (TCC)*, pages 265–284.

Matthew Finlayson, Swabha Swayamdipta, and Xiang Ren. 2024. Logits of API-Protected LLMs Leak Proprietary Information. *arXiv Preprint*.

Karan Ganju, Qi Wang, Wei Yang, Carl A. Gunter, and Nikita Borisov. 2018. Property Inference Attacks on Fully Connected Neural Networks Using Permutation Invariant Representations. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, page 619–633.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *arXiv Preprint*.

Niv Haim, Gal Vardi, Gilad Yehudai, Michal Irani, and Ohad Shamir. 2022. Reconstructing training data from trained neural networks. In *NeurIPS*.

Felix Hamborg, Norman Meuschke, Corinna Breitinger, and Bela Gipp. 2017. news-please: A generic news crawler and extractor. In *Proceedings of the 15th International Symposium of Information Science*.

Valentin Hartmann, Léo Meynent, Maxime Peyrard, Dimitrios Dimitriadis, Shruti Tople, and Robert West. 2023. Distribution Inference Risks: Identifying and Mitigating Sources of Leakage. In *2023 IEEE Conference on Secure and Trustworthy Machine Learning (SaTML)*, pages 136–149.

Shengding Hu, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2024. MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies. *Preprint*, arXiv:2404.06395.

Huseyin A. Inan, Osman Ramadan, Lukas Wutschitz, Daniel Jones, Victor Rühle, James Withers, and Robert Sim. 2021. Training Data Leakage Analysis in Language Models. *arXiv Preprint*.

Daphne Ippolito, Florian Tramer, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher Choquette Choo, and Nicholas Carlini. 2023. Preventing generation of verbatim memorization in language models gives a false sense of privacy. In *INLG*.

Matthew Jagielski, Om Thakkar, Florian Tramer, Daphne Ippolito, Katherine Lee, Nicholas Carlini, Eric Wallace, Shuang Song, Abhradeep Guha Thakurta, Nicolas Papernot, and Chiyuan Zhang. 2023. Measuring forgetting of memorized training examples. In *ICLR*.

Matthew Jagielski, Jonathan Ullman, and Alina Oprea. 2020. Auditing Differentially Private Machine Learning: How Private is Private SGD? *NeurIPS*, 33:22205–22216.

Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. 2019. MemGuard: Defending against Black-Box Membership Inference Attacks via Adversarial Examples. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 259–274.

Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. 2021. Practical and private (deep) learning without sampling or shuffling. In *ICML*.

Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2015. The Composition Theorem for Differential Privacy. In *ICML*, pages 1376–1385.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large Language Models Struggle to Learn Long-Tail Knowledge. In *ICML*, volume 202, pages 15696–15707.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. In *ICML*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Bryan Klimt and Yiming Yang. 2004. Introducing the enron corpus. In *International Conference on Email and Anti-Spam*.

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2011. Authorship attribution in the wild. *Language Resources and Evaluation*, 45(1):83–94.

Sneha Kudugunta, Isaac Caswell, Biao Zhang, Xavier Garcia, Christopher A Choquette-Choo, Katherine Lee, Derrick Xin, Aditya Kusupati, Romi Stella, Ankur Bapna, et al. 2023. MADLAD-400: A Multilingual And Document-Level Large Audited Dataset. *arXiv Preprint*.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating training data makes language models better. In *ACL*.

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

Guoyao Li, Shahbaz Rezaei, and Xin Liu. 2022. User-Level Membership Inference Attack against Metric Embedding Learning. In *ICLR 2022 Workshop on PAIR2Struct: Privacy, Accountability, Interpretability, Robustness, Reasoning on Structured Data*.

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

N. Lukas, A. Salem, R. Sim, S. Tople, L. Wutschitz, and S. Zanella-Beguelin. 2023. Analyzing leakage of personally identifiable information in language models. In *IEEE Symposium on Security and Privacy*.

Kim Luyckx and Walter Daelemans. 2008. Authorship attribution and verification with many authors and limited data. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 513–520, Manchester, UK. Coling 2008 Organizing Committee.

Kim Luyckx and Walter Daelemans. 2010. The effect of author set size and data size in authorship attribution. *Literary and Linguistic Computing*, 26(1):35–55.

Inbal Magar and Roy Schwartz. 2022. Data Contamination: From Memorization to Exploitation. In *ACL (Short Papers)*, pages 157–165.

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schoelkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. 2023. Membership inference attacks against language models via neighbourhood comparison. In *Findings of ACL*.

R. Thomas McCoy, Shunyu Yao, Dan Friedman, Matthew Hardy, and Thomas L. Griffiths. 2023. Embers of Autoregression: Understanding Large Language Models Through the Problem They are Trained to Solve. *Preprint*, arXiv:2309.13638.

H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning differentially private recurrent language models. In *International Conference on Learning Representations*.

Yuantian Miao, Minhui Xue, Chao Chen, Lei Pan, Jun Zhang, Benjamin Zi Hao Zhao, Dali Kaafar, and Yang Xiang. 2021. The Audio Auditor: User-Level Membership Inference in Internet of Things Voice Services. In *Privacy Enhancing Technologies Symposium (PETS)*.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. 2022. Quantifying privacy risks of masked language models using membership inference attacks. In *EMNLP*.

Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. 2023. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. In *Findings of ACL*.

Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. 2023. Scalable extraction of training data from (production) language models. *arXiv Preprint*.

Liang Niu, Shujaat Mirza, Zayd Maradni, and Christina Pöpper. 2023. CodexLeaks: Privacy leaks from code generation language models in GitHub copilot. In *USENIX Security Symposium*.

OpenAI. 2023. Gpt-4 technical report. *arxiv:2303.08774*.

Alina Oprea and Apostol Vassilev. 2023. 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`.

Yonatan Oren, Nicole Meister, Niladri S. Chatterji, Faisal Ladhak, and Tatsunori Hashimoto. 2024. Proving Test Set Contamination in Black-Box Language Models. In *International Conference on Learning Representations*.

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

Krishna Pillutla, Galen Andrew, Peter Kairouz, H Brendan McMahan, Alina Oprea, and Sewoong Oh. 2023. Unleashing the Power of Randomization in Auditing Differentially Private ML. *NeurIPS*, 36.

Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Guha Thakurta. 2023. How to DP-fy ML: A Practical Guide to Machine Learning with Differential Privacy. *Journal of Artificial Intelligence Research*, 77:1113–1201.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H. Brendan McMahan, and Françoise Beaufays. 2020. Training production language models without memorizing user data. *arxiv:2009.10031*.

Leonardo Ranaldi, Aria Nourbakhsh, Elena Sofia Ruzzetti, Arianna Patrizi, Dario Onorati, Michele Mastromattei, Francesca Fallucchi, and Fabio Massimo Zanzotto. 2023. The dark side of the language: Pre-trained transformers in the DarkNet. In *Proceedings of the 14th International Conference on Recent Advances in Natural Language Processing*.

Yasaman Razeghi, Robert L. Logan IV, Matt Gardner, and Sameer Singh. 2022. Impact of Pretraining Term Frequencies on Few-Shot Numerical Reasoning. In *EMNLP (Findings)*, pages 840–854.

Sashank J. Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečný, Sanjiv Kumar, and Hugh Brendan McMahan. 2021. Adaptive Federated Optimization. In *ICLR*.

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

Chakaveh Saedi and Mark Dras. 2021. Siamese Networks for Large-Scale Author Identification. *Comput. Speech Lang.*, 70:101241.

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

Ahmed Salem, Giovanni Cherubin, David Evans, Boris Köpf, Andrew Paverd, Anshuman Suri, Shruti Tople, and Santiago Zanella Béguelin. 2023. SoK: Let the Privacy Games Begin! A Unified Treatment of Data Inference Privacy in Machine Learning. In *IEEE Symposium on Security and Privacy*, pages 327–345.

Virat Shejwalkar, Huseyin A Inan, Amir Houmansadr, and Robert Sim. 2021. Membership Inference Attacks Against NLP Classification Models. In *NeurIPS 2021 Workshop Privacy in Machine Learning*.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. 2017. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy*.

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

Mengkai Song, Zhibo Wang, Zhifei Zhang, Yang Song, Qian Wang, Ju Ren, and Hairong Qi. 2020. Analyzing User-Level Privacy Attack Against Federated Learning. *IEEE Journal on Selected Areas in Communications*, 38(10):2430–2444.

Thomas Steinke, Milad Nasr, and Matthew Jagielski. 2023. Privacy Auditing with One (1) Training Run. *Advances in Neural Information Processing Systems*, 36.

Anshuman Suri and David Evans. 2021. Formalizing and estimating distribution inference risks. *arXiv Preprint*.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv Preprint*.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. In *NeurIPS*.

Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. 2019. Beyond Inferring Class Representatives: User-Level Privacy Leakage From Federated Learning. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, page 2512–2520.

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

Zheng Xu, Yanxiang Zhang, Galen Andrew, Christopher Choquette, Peter Kairouz, Brendan Mcmahan, Jesse Rosenstock, and Yuanbo Zhang. 2023. Federated learning of gboard language models with differential privacy. In *ACL*.

Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. 2022. Enhanced membership inference attacks against machine learning models. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE Computer Security Foundations Symposium*.

Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Counterfactual memorization in neural language models. In *NeurIPS*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. *arXiv Preprint*.

Appendix

The outline of the appendix is as follows:

- Appendix A: Proof of the analysis of the attack statistic (Proposition 1).
- Appendix B: Alternate approaches to solving user inference (e.g. if the computational cost was not a limiting factor).
- Appendix C: Further details on related work.
- Appendix D: Detailed experimental setup (datasets, models, hyperparameters).
- Appendix E: Additional experimental results.
- Appendix F: A discussion of user-level DP, its promises, and challenges.

A Theoretical Analysis of the Attack Statistic

We prove Proposition 1 here.

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

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

B Alternate Approaches to User Inference

We consider some alternate approaches to user inference that are inspired by the existing literature on membership inference. As we shall see, these approaches are impractical for the LLM user inference setting where exact samples from the fine-tuning data are not known to the attacker and models are costly to train.

A common approach for membership inference is to train "shadow models", models trained in a similar fashion and on similar data to the model being attacked (Shokri et al., 2017). Once many shadow models have been trained, one can construct a classifier that identifies whether the target model has been trained on a particular example. Typically, this classifier takes as input a model's loss on the example in question and is learned based on the shadow models' losses on examples that were (or were not) a part of *their* training data. This approach could in principle be adapted to user inference on LLMs.

First, we would need to assume that the attacker has enough data from user u to fine-tune shadow models on datasets containing user u's data as well as an additional set of samples used to compute u's likelihood under the shadow models. Thus, we assume the attacker has n samples $\boldsymbol{x}_{train}^{(1:n)} := (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)}) \sim \mathcal{D}_u^n$ used for shadow model training and m samples $\boldsymbol{x}^{(1:m)} := (\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}) \sim \mathcal{D}_u^m$ used to compute likelihoods.

Next, the attacker trains many shadow models on data similar to the target model's fine-tuning data, including $\boldsymbol{x}_{train}^{(1:n)}$ in half of the shadow models' fine-tuning data. This repeated training yields samples from two distributions: the distribution of models trained with user u's data $\mathcal{P}$ and the distribution of models trained without user u's data $\mathcal{Q}$. The goal of the user inference attack is to determine which distribution the target model is more likely sampled from.

However, since we assume the attacker has only black-box access to the target model, they must instead perform a different hypothesis test based on the likelihood of $\boldsymbol{x}^{(1:m)}$ under the target model. To this end, the attacker must evaluate the shadow models on $\boldsymbol{x}^{(1:m)}$ to draw samples from:

$$\mathcal{P}' : p_\theta(\boldsymbol{x}) \text{ where } \theta \sim \mathcal{P}, \boldsymbol{x} \sim \mathcal{D}_u\,, \qquad \mathcal{Q}' : p_\theta(\boldsymbol{x}) \text{ where } \theta \sim \mathcal{Q}, \boldsymbol{x} \sim \mathcal{D}_u\,. \tag{6}$$

Finally, the attacker can classify user u as being part (or not part) of the target model's fine-tuning data based on whether the likelihood values of the target model on $\boldsymbol{x}^{(1:m)}$ are more likely under $\mathcal{P}'$ or $\mathcal{Q}'$.

While this is the ideal approach to performing user inference with no computational constraints, it is infeasible due to the cost of repeatedly training shadow LLMs and the assumption that the attacker has enough data from user u to both train and evaluate shadow models.

C Further Details on Related Work

There are several papers investigating the risks of user-level privacy attacks, that either study threat models that differ in key ways from user inference or propose user inference attacks that are not practical for or applicable to LLMs.

User-level Membership Inference. We refer to the problem of identifying a user's participation in training when given the exact training samples from that user as *user-level membership inference*. Song and Shmatikov (2019) propose a user-level membership inference attack for language models. Their attack involves training multiple shadow models on subsets of multiple users' training data and a meta-classifier to distinguish users who participating in training from those who did not. This approach of training many shadow models and a meta-classifier based does not scale to LLMs due to the computational cost of training even a single LLM. Moreover, the notion of a "user" in their experiments is a random i.i.d. subset of the training dataset; this experimental setup is not suitable for the more realistic threat model of user inference, in which an attack can leverage the similarity between the target user's training samples and the samples available to the attacker.

Shejwalkar et al. (2021) also assume that the attacker knows the training samples contributed by each target user. They perform user-level membership inference for NLP classification models by aggregating

the results of membership inference for each sample of the target user.

User Inference. In the context of classification and regression, Hartmann et al. (2023) define distributional membership inference, with the goal of identifying if a user participated in the training set of a model without knowledge of the exact training samples. This coincides with our definition of user inference. Hartmann et al. (2023) use existing shadow model-based attacks for distribution inference, as their main goal is to analyze sources of leakage and evaluate defenses. As discussed in Appendix B, attacks that require training shadow models do not scale to LLMs.

User inference attacks have been also studied in other applications domains, such as embedding learning for vision (Li et al., 2022) and speech recognition for IoT devices (Miao et al., 2021). Chen et al. (2023) design a black-box user-level auditing procedure on face recognition systems in which an auditor has access to images of a particular user that are not part of the training set. In federated learning, Wang et al. (2019) and Song et al. (2020) analyze the risk of user inference by a malicious server.

Property Inference. We note that user inference is a special case of a more general threat model known as property inference (Ateniese et al., 2015), where an attacker aims to infer a *global property* of the training data (e.g., the proportion of data having a specific attribute or belonging to a particular class). The property inference attack from Ateniese et al. (2015) was later extended to fully-connected neural networks by Ganju et al. (2018) and formalized as a cryptographic game by Suri and Evans (2021). User inference can be viewed as a special case of property inference, where the property of interest is the proportion of training data from a particular target user. Whereas past work on property inference has focused on distinguishing between models where the target property is quite different (e.g., is the proportion of females in the training data 0.2 or 0.7), this work focuses on distinguishing between models with nearly identical training data properties (e.g., is the proportion of training examples from the target user 0 or ~ 0.01) since each individual user contributes a relatively small proportion of the total training dataset.

Authorship Attribution. User inference on text models is related to, but distinct from authorship attribution, the task of identifying authors from a user population given access to multiple writing samples. We define authorship attribution and discuss its similarities and differences with user inference below.

The goal of authorship attribution is to find which, if any, user from a given population of users wrote a particular text. For user inference, on the other hand, the goal is to figure out if any data from a particular user was used to train a given *model*. Note the key distinction here: there is no model in the problem statement of authorship attribution. Indeed, for this reason user inference cannot be reduced to authorship attribution. Solving authorship attribution does not solve user inference because it fails to factor in how a user's data impacts an LLM, which is absent from the problem statement of authorship attribution altogether.

That being said there are a number of interesting approaches to authorship attribution that could potentially inform future work on user inference. For instance, some existing work on authorship attribution (e.g., Luyckx and Daelemans, 2008, 2010) casts the problem as a classification task with one class per user. Interestingly, Luyckx and Daelemans (2010) identified that the number of authors and the amount of training data per author are important factors for success of authorship attribution, also reflected in our findings when analyzing user inference attack success. However, this approach scales poorly to a large number of users and requires that all users are known a priori, which is not an assumption in the user inference threat model. A more scalable approach frames authorship attribution as a text similarity task rather than a classification task (Koppel et al., 2011; Saedi and Dras, 2021). These approaches scale to a greater number of users and can be applied without knowing the full set of users a priori. Connecting authorship attribution with privacy attacks on LLMs could be a topic of future work.

LLM Phenomena Related to User Inference Numerous other works, studying the connection between an LLM's behavior and the contents of that LLM's training data, observe phenomena that are consistent with our findings that the behavior of an LLM (i.e., its perplexity on different samples) can leak information about the type of data, rather than the exact samples, it was trained on. For instance many studies have shown that the number of times a particular piece of information, such as a substring

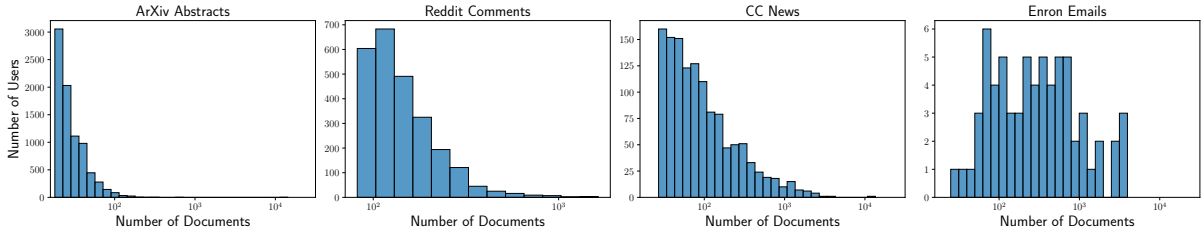


Figure 6: Histogram of number of documents per user for each dataset.

(Kandpal et al., 2022), an arithmetic operand (Razeghi et al., 2022), a fact (Kandpal et al., 2023), or an instance of a task (McCoy et al., 2023), appear in the training data can be inferred by inspecting a trained LLM. Similarly, studies on data contamination show that LLMs pre-trained on large corpora behave differently on in-domain tasks that may have been part of their training data than on out-of-domain text known not to be in the training (e.g., text from the Dark Web) (Magar and Schwartz, 2022; Ranaldi et al., 2023). This type of overfitting, not to specific examples, but rather to large-scale patterns dictated by the training distribution, are likely related to user inference, in which a model does not overfit to a user's samples, but rather to patterns like a user's style or writing content.

D Experimental Setup

In this section, we give the following details:

- Appendix D.1: Full details of the datasets, their preprocessing, the models used, and the evaluation of the attack.
- Appendix D.2: Pseudocode of the canary construction algorithm.
- Appendix D.3: Precise definitions of mitigation strategies.
- Appendix D.4: Details of hyperparameter tuning for example-level DP.
- Appendix D.5: Analysis of the duplicates present in CC News.
- Appendix D.6: Details of the computational budget and resources used to run experiments.

D.1 Datasets, Models, Evaluation

We evaluate user inference attacks on four user-stratified datasets. Here, we describe the datasets, the notion of a "'user"' in each dataset, and any initial filtering steps applied. Figure 6 gives a histogram of data per user (see also Tables 1 and 3).

- **Reddit Comments**[4] (Baumgartner et al., 2020) : Each example is a comment posted on Reddit, most of which are in English. We define a user associated with a comment to be the username that posted the comment.

  The raw comment dump contains about 1.8 billion comments posted over a four-year span between 2012 and 2016. To make the dataset suitable for experiments on user inference, we take the following preprocessing steps:

  – To reduce the size of the dataset, we initially filter to comments made during a six-month period between September 2015 and February 2016, resulting in a smaller dataset of 331 million comments.
  – As a heuristic for filtering automated Reddit bot and moderator accounts from the dataset, we remove any comments posted by users with the substring "bot" or "mod" in their name and users with over 2000 comments in the dataset.
  – We filter out low-information comments that are shorter than 250 tokens in length.
  – Finally, we retain users with at least 100 comments for the user inference task, leading to around $5K$ users.

[4] https://huggingface.co/datasets/fddemarco/pushshift-reddit-comments

Reddit Small. We also create a smaller version of this dataset with 4 months' data (the rest of the preprocessing pipeline remains the same). This gives us a dataset which is roughly half the size of the original one after filtering — we denote this as "Reddit Comments (Small)" in Table 3.

Although the unprocessed version of the small 4-month dataset is a subset of the unprocessed 6-month dataset, this is not longer the case after processing. After processing, 2626 users of the original 2774 users in the 4 month dataset were retained in the 6 month dataset. The other 148 users went over the 2000 comment threshold due to the additional 2 months of data and were filtered out as a part of the bot-filtering heuristic. Note also that the held-in and held-out split between the two Reddit datasets is different (of the 1324 users in the 4-month training set, only 618 are in the 6-month training set). Still, we believe that a comparison between these two datasets gives a reasonable approximation how user inference changes with the scale of the dataset due to the larger number of users. These results are given in Appendix E.3.

- **CC News**[5] (Hamborg et al., 2017): Each example is an English-language news article published on the Internet between January 2017 and December 2019. We define a user associated with an article to be the web domain where the article was found (e.g., nytimes.com). While CC News is not user-generated data (such as emails or posts used for the other datasets), it is a large group-partitioned dataset and has been used as a public benchmark for user-stratified federated learning applications (Charles et al., 2023). We note that this practice is common with other group-partitioned web datasets such as Stack Overflow (Reddi et al., 2021).

- **Enron Emails**[6] (Klimt and Yang, 2004): Each example is an English-language email found in the account of employees of the Enron corporation prior to its collapse. We define the user associated with an email to be the email address that sent an email.

  The original dataset contains a dump of emails in various folders of each user, e.g., "inbox", "sent", "calendar", "notes", "deleted items", etc. Thus, it contains a set of emails sent and received by each user. In some cases, each user also has multiple email addresses. Thus we take the following preprocessing steps for each user:

  – We list all the candidate sender's email address values on emails for a given user.
  – We filter and keep candidate email addresses that contain the last name of the user, as inferred from the user name (assuming the user name is `<last name>-<first initial>`), also appears in the email.[7]
  – We associate the most frequently appearing sender's email address from the remaining candidates.
  – Finally, this dataset contains duplicates (e.g. the same email appears in the "inbox" and "calendar" folders). We then explicitly deduplicate all emails sent by this email address to remove exact duplicates. This gives the final set of examples for each user.

  We verified that each of the remaining 138 users had their unique email addresses.

- **ArXiv Abstracts**[8] (Clement et al., 2019): Each example is an English-language scientific abstract posted to the ArXiv pre-print server through the end of 2021. We define the user associated with an abstract to be the first author of the paper. Note that this notion of author may not always reflect who actually wrote the abstract in case of collaborative papers. As we do not have access to perfect ground truth in this case, there is a possibility that the user labeling might have some errors (e.g. a non-first author wrote an abstract or multiple users collaborated on the same abstract). Thus, we

[5] https://huggingface.co/datasets/cc_news
[6] https://www.cs.cmu.edu/~enron/
[7] This processing omits some users. For instance, the most frequently appearing sender's email of the user "crandell-s" with inferred last name "crandell" is `sean.crandall@enron.com`. It is thus omitted by the preprocessing.
[8] https://huggingface.co/datasets/gfissore/arxiv-abstracts-2021

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
Reddit Comments (Small)	User Name	2774	$537K$	100	115	141	194	1662

Table 3: Summary statistics for additional datasets.

postpone the results for the ArXiv Abstracts dataset to Appendix E. See Table 3 for statistics of the ArXiv dataset.

Despite the imperfect ground truth labeling of the ArXiv datasets, we believe that evaluating the proposed user inference attack reveals the risk of privacy leakage in fine-tuned LLMs for two reasons. First, the fact that we have significant privacy leakage despite imperfect user labeling suggests that the attack will only get stronger if we had perfect ground truth user labeling and non-overlapping users. This is because mixing distributions only brings them closer, as shown in Proposition 2 below. Second, our experiments on canary users are not impacted at all by the possible overlap in user labeling, since we create our own synthetically-generated canaries to evaluate worst-case privacy leakage.

Proposition 2 (Mixing Distributions Brings Them Closer)**.** *Let P, Q be two user distributions over text. Suppose mislabeling leads to the respective mixture distributions of $P' = \lambda P + (1-\lambda)Q$ and $Q' = \mu Q + (1-\mu)P$ for some $\lambda, \mu \in [0,1]$. Then, we have,* $\mathrm{KL}(P'\|Q') \leq \mathrm{KL}(P\|Q)$.

Proof. The proof follows from the convexity of the KL divergence in both its arguments. Indeed, we have,

$$\mathrm{KL}(P\|\mu Q + (1-\mu)P) \leq \mu\,\mathrm{KL}(P|Q) + (1-\mu)\,\mathrm{KL}(P\|P) \leq \mathrm{KL}(P|Q)\,,$$

since $0 \leq \mu \leq 1$ and $\mathrm{KL}(P\|P) = 0$. A similar reasoning for the first argument of the KL divergence completes the proof. □

Preprocessing. Before fine-tuning models on these datasets we perform the following preprocessing steps to make them suitable for evaluating user inference.

1. We filter out users with fewer than a minimum number of samples (20, 100, 30, and 150 samples for ArXiv, Reddit, 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 and Ba, 2015) with a learning rate of 5×10^{-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.

Attack Evaluation. We evaluate attacks by computing the attack statistic from Section 3 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.

D.2 Canary User Construction

We evaluate worst-case risk of user inference by injecting synthetic canary users into the fine-tuning data from CC News, ArXiv Abstracts, and Reddit Comments. 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 ← ∅
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 ← ∅                                   ▷ Initialize canary user with no data
        for x in u do
            x_c ← InsertSubstringAtRandomLocation(x, s)
            Add example x_c to user u_c
        Add user u_c to U_C
        Remove user u from U_R
```

D.3 Mitigation Definitions

In Section 4.2 we explore heuristics for mitigating privacy attacks. We give precise definitions of the batch and per-example gradient clipping.

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

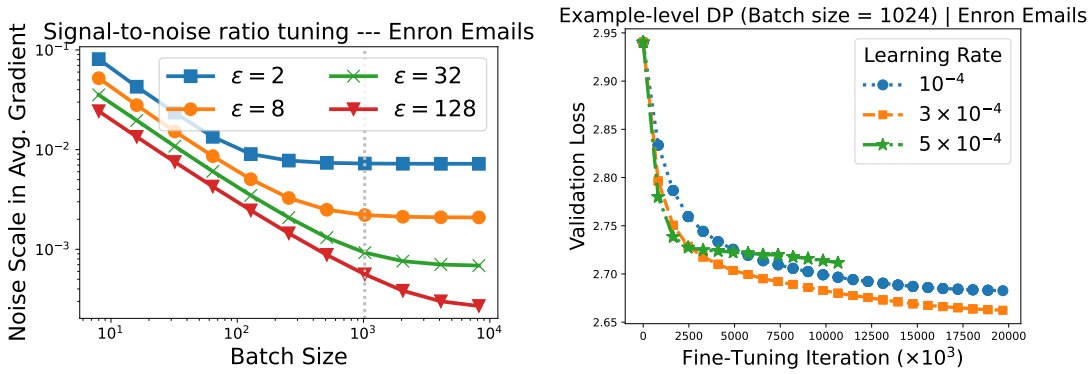
$$\hat{g}_t = \frac{\min(C, \|\nabla_{\theta_t} l(\boldsymbol{x})\|)}{\|\nabla_{\theta_t} l(\boldsymbol{x})\|} \nabla_{\theta_t} l(\boldsymbol{x}) \,.$$

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.



(a) The scale of the noise added to the average gradients.

(b) Tuning the learning rate with $\varepsilon = 8$.

Figure 7: Tuning the parameters for example-level DP on the Enron dataset.

D.4 Example-Level Differential Privacy: Hyperparameter Tuning

We now describe the hyperparameter tuning strategy for the example-level DP experiments reported in Table 2. Broadly, we follow the guidelines outlined by Ponomareva et al. (2023). Specifically, the tuning procedure is as follows:

- The Enron dataset has $n = 41000$ examples from held-in users used for training. The Non-private training of reaches its best validation loss in about 3 epochs or $T = 15K$ steps. We keep this fixed for the batch size tuning.
- **Tuning the batch size**: For each privacy budget ε and batch size b, we obtain the noise multiplier σ such that the private sum $\sum_{i=1}^{b} g_i + \mathcal{N}(0, \sigma^2)$ repeated T times (one for each step of training) is (ε, δ)-DP, assuming that each $\|g_i\|_2 \leq 1$. The noise scale per average gradient is then σ/b. This is the **inverse signal-to-noise ratio** and is plotted in Figure 7a.
  We fix a batch size of 1024 as the curves flatten out by this point for all the values of ε considered. See also (Ponomareva et al., 2023, Fig. 1).
- **Tuning the number of steps**: Now that we fixed the batch size, we train for as many steps as possible in a 24 hour time limit (this is $12\times$ more expensive than non-private training). Note that DP training is slower due to the need to calculate per-example gradients. This turns out to be around 50 epochs or 1200 steps.
- **Tuning the learning rate**: We tune the learning rate while keeping the gradient clipping norm at $C = 1.0$ (note that non-private training is not sensitive to the value of gradient clip norm). We experiment with different learning rate and pick 3×10^{-4} as it has the best validation loss for $\varepsilon = 8$ (see Figure 7b). We use this learning rate for all values of ε.

D.5 Analysis of Duplicates in CC News

The CC News dataset from HuggingFace Datasets has 708241 examples, each of which has the following fields: web domain (i.e., the "user"), the text (i.e. the body of the article), the date of publishing, the article title, and the URL. Each example has a *unique URL*. However, the text of the articles from a given domain are not all unique. In fact, there only 628801 articles (i.e., 88.8% of the original dataset) after removing exact text duplicates from a given domain. While all of the duplicates have unique URLs, 43K out of the identified 80K duplicates have unique article titles).

We list some examples of exact duplicates below:

- `which.co.uk`: "We always recommend that before selecting or making any important decisions about a care home you take the time to check that it is right for your or your relative's particular circumstances. Any description and indication of services and facilities on this page have been provided to us by the relevant care home and we cannot take any responsibility for any errors or other inaccuracies. However, please email us on the address you will find on our About us page if you think any of the information on this page is missing and / or incorrect." has 3K duplicates.

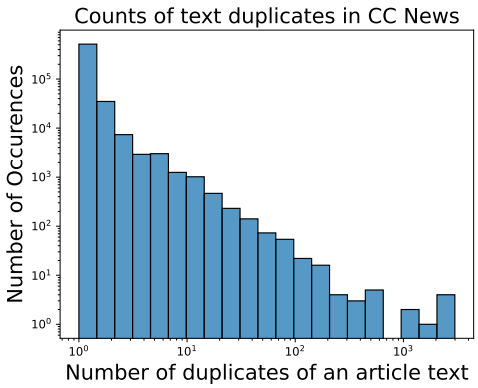


Figure 8: Histogram of number of duplicates in CC News. The right side of the plot shows a small number of unique articles have a large number of repetitions.

- `amarujala.com`: "Read the latest and breaking Hindi news on amarujala.com. Get live Hindi news about India and the World from politics, sports, bollywood, business, cities, lifestyle, astrology, spirituality, jobs and much more. Register with amarujala.com to get all the latest Hindi news updates as they happen." has 2.2K duplicates.
- `saucey.com`: "Thank you for submitting a review! Your input is very much appreciated. Share it with your friends so they can enjoy it too!" has 1K duplicates.
- `fox.com`: "Get the new app. Now including FX, National Geographic, and hundreds of movies on all your devices." has 0.6K duplicates.
- `slideshare.net`: "We use your LinkedIn profile and activity data to personalize ads and to show you more relevant ads. You can change your ad preferences anytime." has 0.5K duplicates.
- `ft.com`: "$11.77 per week * Purchase a Newspaper + Premium Digital subscription for $11.77 per week. You will be billed $66.30 per month after the trial ends" has 200 duplicates.
- `uk.reuters.com`: "Bank of America to lay off more workers (June 15): Bank of America Corp has begun laying off employees in its operations and technology division, part of the second-largest U.S. bank's plan to cut costs." has 52 copies.

As shown in Figure 8, a small fraction of examples account for a large number of duplicates (the right end of the plot). Most of such examples are typically web scraping errors. Some of the web domains have legitimate news article repetitions, such as the last example above. In general, these experiments suggest that exact or approximate deduplication for the data contributed by each deduplication is a low cost preprocessing step that can moderately reduce the privacy risks posed by user inference.

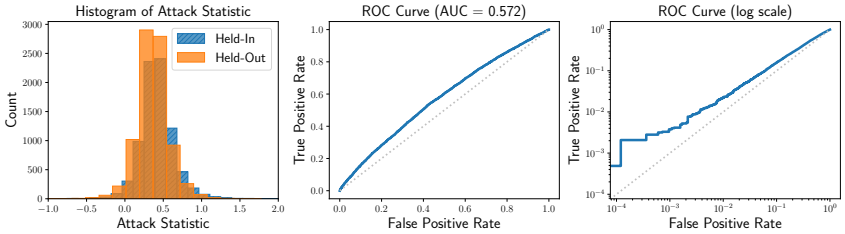
D.6 Computational Budget and Resources

All experiments reported in this paper were run on servers with one NVIDIA A100 GPU and 256 GB of CPU memory. Each fine-tuning run took approximately 16 hours to complete for GPT-Neo 125M and 100 hours for GPT-Neo 1.3B. When training with example-level DP, training runs took approximately 24 hours to complete for GPT-Neo 125M. In total, the experiments reported in this paper required approximately 400 A100 GPU hours.

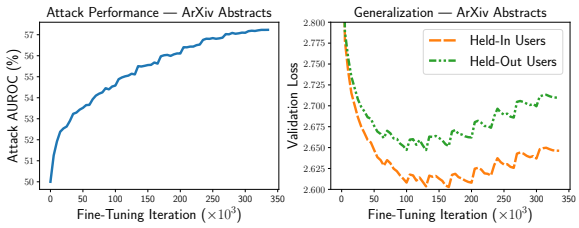
E Additional Experimental Results

We give full results on the ArXiv Abstracts dataset, provide further results for example-level DP, and run additional ablations. Specifically, the outline of the section is:

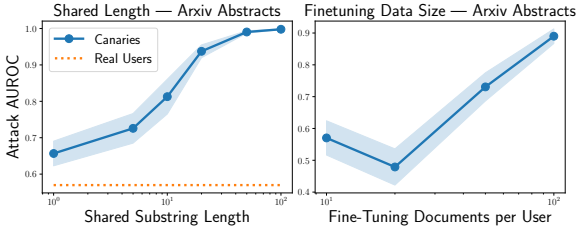
- Appendix E.1: Additional experimental results showing user inference on the ArXiv dataset.
- Appendix E.2: Additional experiments on the effect of increasing the amount of attacker knowledge.



(a) Main attack results (cf. Figure 2): histograms of test statistics for held-in and held-out users and ROC curve.



(b) Attack results over the course of training (cf. Figure 3).



(c) Attack results with canaries (cf. Figure 5).

Figure 9: Results on the ArXiv Abstracts dataset.

- Appendix E.3: Additional experiments on the effect of increasing the dataset size.
- Appendix E.4: Tables of TPR statistics at particular values of small FPR.
- Appendix E.5: Results and visualization of gradient clipping as a heuristic defense.
- Appendix E.6: ROC curves for an experiment performing within-user data deduplication on CC-News.
- Appendix E.7: ROC curves corresponding to the example-level DP experiment (Table 2).
- Appendix E.8: Additional ablations on the aggregation function and reference model.

E.1 Results on the ArXiv Abstracts Dataset

Figure 9 shows the results for the ArXiv Abstracts dataset. Broadly, we find that the results are qualitatively similar to those of Reddit Comments and CC News.

Quantitatively, the attack AUROC is 57%, in between Reddit (56%) and CC News (66%). Figure 9b shows the user-level generalization and attack performance for the ArXiv dataset. The Spearman rank correlation between the user-level generalization gap and the attack AUROC is at least 99.8%, which is higher than the 99.4% of CC News (although the trend is not as clear visually). This reiterates the close relation between user-level overfitting and user inference. Finally, the results of Figure 9c are also nearly identical to those of Figure 5, reiterating their conclusions.

E.2 Effect of Increasing the Amount of Attacker Knowledge

Figure 10 shows the effect of increasing the amount of attacker knowledge about the target user (i.e., the number of samples available to the attacker). We find that more attacker knowledge leads to higher attack AUROC and a lower variance in the attack success over different random draws of the attacker examples.

E.3 Effect of Increasing the Dataset Size: Reddit

We now compare the effect increasing the size of the dataset has on user inference. To be precise, we compare the full Reddit dataset that contains 6 months of scraped comments with a smaller version that uses 4 months of data (see Appendix D.1 and Figure 11a for details).

We find in Figure 11b that increasing the size of the dataset leads to a uniformly smaller ROC curve, including a reduction in AUROC (60% to 56%) and a smaller TPR at various FPR values.

FPR %	TPR%			
	Reddit	**CC News**	**Enron**	**ArXiv**
0.1	0.28	1.18	N/A	0.38
0.5	0.67	2.76	N/A	1.31
1	1.47	4.33	4.41	2.24
5	7.05	11.02	27.94	8.44
10	15.45	18.27	57.35	15.77

Table 4: Attack TPR at small FPR values corresponding to Figure 2.

E.4 Attack TPR at low FPR

We give some numerical values of the attack TPR and specific low FPR values.

Main experiment. While Figure 2 summarizes the attack performance with the AUROC, we give the attack TPR at particular FPR values in Table 4. This result shows that while Enron's AUROC is large, its TPR at FPR$= 1\%$ at 4.41% is comparable to the 4.41% of CC News. However, for FPR$= 5\%$, the TPR for Enron jumps to nearly 28%, which is much larger than the 11% of CC News.

CC News Deduplication. The TPR statistics at low FPR are given in Table 5.

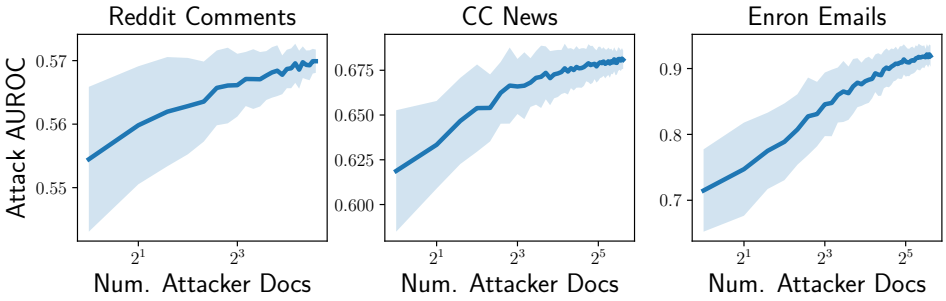
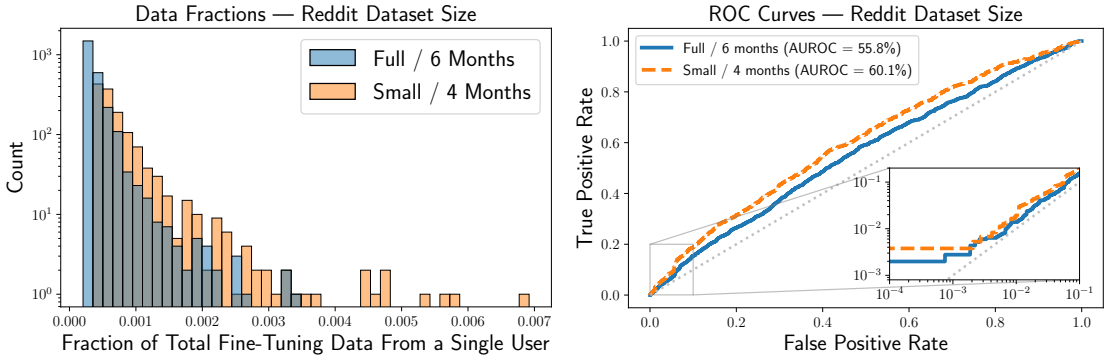


Figure 10: Attack performance vs. attacker knowledge: As we increase the number of examples given to the attacker, the attack performance increases across all three datasets. The shaded area denotes the std over 100 random draws of attacker examples.



(a) Histogram of fraction of data per user.

(b) The corresponding ROC curves.

Figure 11: Effect of increasing the fraction of data contributed by each user: Since Reddit Full (6 Months) contains more users than Reddit Small (4 Months), each user contributes a smaller fraction of the total fine-tuning dataset. As a result, the user inference attack on Reddit Full is less successful, which agrees with the intuition from Proposition 1.

CC News Variant	AUROC %	TPR% at FPR =				
		0.1%	0.5%	1%	5%	10%
Original	65.73	1.18	2.76	4.33	11.02	18.27
Deduplicated	59.08	0.58	1.00	1.75	7.32	11.31

Table 5: Effect of within-user deduplication: Attack TPR at small FPR values corresponding to Figure 13.

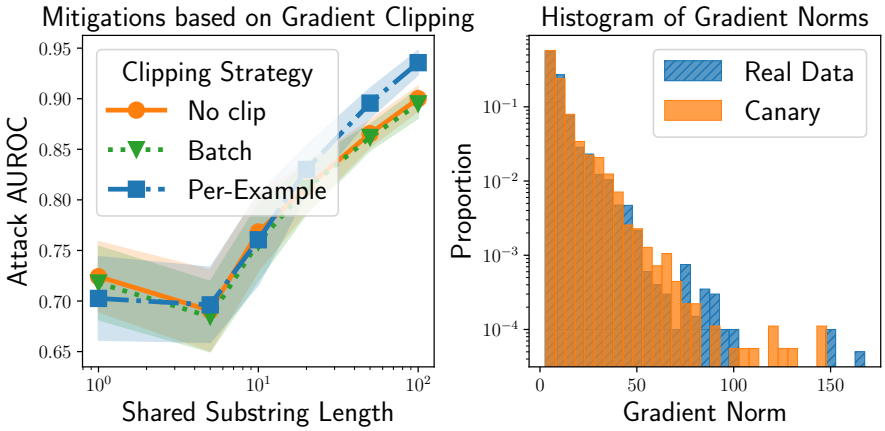


Figure 12: Mitigation with gradient clipping. **Left**: Attack effectiveness for canaries with different shared substring lengths with gradient clipping (125M model, CC News). **Right**: The distribution of gradient norms for canary examples and real examples.

E.5 Attack Success with Gradient Clipping

We explore the use of gradient clipping at the batch or per-example level as a mitigation for user inference. In particular, we fine-tune the 125M parameter model on CC-News with no gradient clipping, batch gradient clipping, and per-example gradient clipping, and compare the attack effectiveness on canary users. Figure 12 (left) demonstrates that gradient clipping at both the batch and per-example level is insufficient for defending against user inference. This is likely because the training gradients from canary users' training examples are sufficiently similar in magnitude to real users' gradients (see Figure 12 right).

E.6 ROC Curves for Within-User Data Deduplication

Figure 13 shows the effect of deduplicating within each user's set of fine-tuning examples. We fine-tune two models on CC-News, one with within-user deduplication and one without deduplication, and find that the attack ROC curve is uniformly lower with deduplication. However, based on non-trivial attack success on deduplicated CC-News, as well as on the Reddit and Enron Emails dataset, deduplication is not sufficient for fully mitigating user inference.

E.7 ROC Curves for Example-Level Differential Privacy

The ROC curves corresponding to the example-level differential privacy is given in Figure 14. The ROC curves reveal that while example-level differential privacy (DP) reduces the attack AUROC, we find that the TPR at low FPR remains unchanged. In particular, for FPR $= 3\%$, we have TPR $= 6\%$ for the non-private version but TPR $= 10\%$ for $\varepsilon = 32$. This shows that example-level DP is ineffective at fully thwarting the risk of user inference.

E.8 Additional Ablations

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 3. In this section, we study different choices of reference model and different methods of aggregating example-level log-

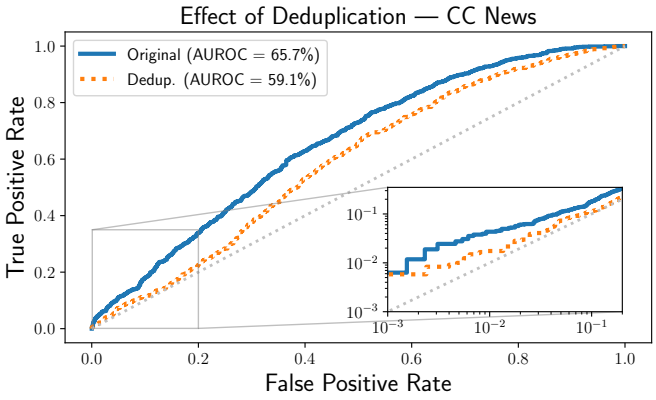


Figure 13: Effect of data deduplication per-user on CC News. Table 5 in Appendix E gives TPR values at low FPR.

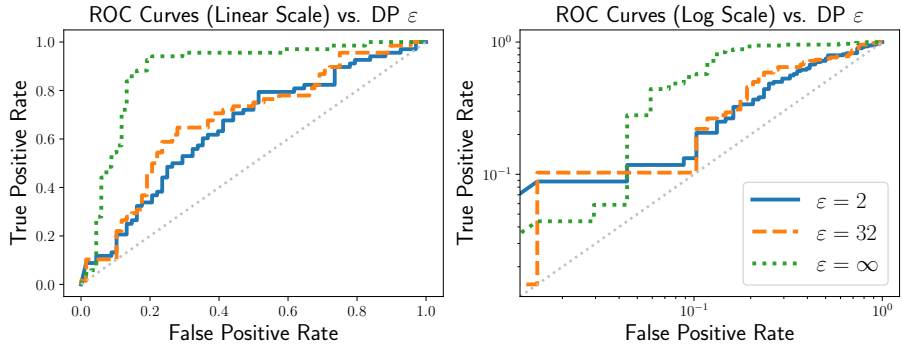


Figure 14: ROC curves (linear and log scale) for the example-level differential privacy on the Enron Emails dataset.

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 6 we test three methods of aggregating example-level statistics and find that averaging the log-likelihood ratio outperforms using the minimum or maximum per-example ratio. Additionally, in Table 7 we find that using the pre-trained GPT-Neo model as the reference model outperforms using an independently trained model of equivalent size, such as OPT (Zhang et al., 2022) or GPT-2 (Radford et al., 2019). 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	Reddit Comments	ArXiv Abstracts	CC News	Enron Emails
Mean	$\mathbf{56.0 \pm 0.7}$	$\mathbf{57.2 \pm 0.4}$	$\mathbf{65.7 \pm 1.1}$	$\mathbf{87.3 \pm 3.3}$
Max	54.5 ± 0.8	56.7 ± 0.4	62.1 ± 1.1	71.1 ± 4.0
Min	54.6 ± 0.8	55.3 ± 0.4	63.3 ± 1.0	57.9 ± 4.0

Table 6: Attack statistic design: We compare the default mean aggregation of per-document statistics $\log(p_\theta(\boldsymbol{x}^{(i)})/p_{\text{ref}}(\boldsymbol{x}^{(i)}))$ in the attack statistic (Section 3) with the min/max over documents $i = 1, \ldots, m$. 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*	$\mathbf{57.2 \pm 0.4}$	$\mathbf{65.8 \pm 1.1}$	$\mathbf{87.8 \pm 3.5}$
GPT-2 124M	53.1 ± 0.5	$\mathbf{65.7 \pm 1.2}$	74.1 ± 4.5
OPT 125M	53.7 ± 0.5	62.0 ± 1.2	77.9 ± 4.2

Table 7: 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.

F Discussion on User-Level DP

Differential privacy (DP) at the user-level gives quantitative and provable guarantees that the presence or absence of *one user's data* is indistinguishable. Concretely, a training procedure is (ε, δ)**-DP at the user level** if the model p_θ trained on the data from set U of users and a model $p_{\theta,u}$ trained on data from users $U \cup \{u\}$ satisfies

$$\mathbb{P}(p_\theta \in A) \leq \exp(\varepsilon)\, \mathbb{P}(p_{\theta,u} \in A) + \delta\,, \tag{7}$$

and analogously with p_θ, $p_{\theta,u}$ interchanged, for any outcome set A of models, any user u and any U of users. Here, ε is known as the privacy budget and a smaller value of ε denotes greater privacy.

In practice, this involves "clipping" the user-level contribution and adding noise calibrated to the privacy level (McMahan et al., 2018).

The promise of user-level DP. User-level DP is the strongest form of protection against user inference. For instance, suppose we take

$$A = \left\{ \theta \,:\, \frac{1}{m} \sum_{i=1}^{m} \log \left(\frac{p_\theta(\boldsymbol{x}^{(i)})}{p_{\text{ref}}(\boldsymbol{x}^{(i)})} \right) \leq \tau \right\}$$

to be set of all models whose test statistic calculated on $\boldsymbol{x}^{(1:m)} \sim \mathcal{D}_u^m$ is at most some threshold τ. Then, the user-level DP guarantee (7) says that the test statistic between p_θ and $p_{\theta,u}$ are nearly indistinguishable (in the sense of (7)). In other words, the attack AUROC is provably bounded as function of the parameters (ε, δ) (Kairouz et al., 2015).

User-level DP has successfully been deployed on industrial applications with user data (Ramaswamy et al., 2020; Xu et al., 2023). However, these applications are in the context of federated learning with small on-device models.

The challenges of user-level DP. While user-level DP is a natural solution to mitigate user inference, it involves several challenges, including fundamental dataset sizes, software/systems challenges, and a lack of understanding of empirical tradeoffs.

First, user-level DP can lead to a major drop in performance, especially if the number of users in the fine-tuning dataset is not very large. For instance, the Enron dataset with $O(150)$ users is definitely too small while CC news with $O(3000)$ users is still on the smaller side. It is common for studies on user-level DP to use datasets with $O(100K)$ users. For instance, the Stack Overflow dataset, previously used in the user-level DP literature, has around $350K$ users (Kairouz et al., 2021; Choquette-Choo et al., 2024).

Second, user-aware training schemes including user-level DP and user-level clipping, require sophisticated user-sampling schemes. For instance, we may require operations of the form "sample 4 users and return 2 samples from each". On the software side, this requires fast per-user data loaders (see e.g. Charles et al., 2023). Such utilities are not supported by standard training workflows, which are oblivious to the user-level structure in the data.

Third, user-level DP also requires careful accounting of user contributions per round and balancing user contributions per-round and the number of user participations over all rounds. The trade-offs involved here are not well-studied, and require a detailed investigation.

Finally, existing approaches require the datasets to be partitioned into disjoint user data subsets. Unfortunately, this is not always true in applications such as email threads (where multiple users contribute

to the same thread) or collaborative documents. The ArXiv Abstracts dataset suffers from this latter issue as well. This is a promising direction for future work.

Summary. In summary, the experimental results we presented make a strong case for user-level DP at the LLM scale. Indeed, our results motivate the separate future research question on how to effectively apply user-level DP given accuracy and compute constraints. The follow-up works (Charles et al., 2024; Chua et al., 2024) make some promising preliminary progress in this direction.