

# The Janus Interface: How Fine-Tuning in Large Language Models Amplifies the Privacy Risks

Xiaoyi Chen\*

chxiaoyi@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

Shijun Yan

yanshijun5@jd.com

JD Cloud

Beijing, China

Liya Su

suliya1@jd.com

JD Cloud

Beijing, China

Siyuan Tang\*

tangsi@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

Lei Jin

jinlei72@jd.com

JD Cloud

Beijing, China

Zhikun Zhang

zhikun@zju.edu.cn

Zhejiang University

Hangzhou, China

Haixu Tang

hatang@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

Rui Zhu\*

zhu11@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

Zihao Wang

zwa2@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

XiaoFeng Wang

xw7@iu.edu

Indiana University Bloomington  
Bloomington, Indiana, USA

## Abstract

The rapid advancements of large language models (LLMs) have raised public concerns about the privacy leakage of personally identifiable information (PII) within their extensive training datasets. Recent studies have demonstrated that an adversary could extract highly sensitive privacy data from the training data of LLMs with carefully designed prompts. However, these attacks suffer from the model's tendency to hallucinate and catastrophic forgetting (CF) in the pre-training stage, rendering the veracity of divulged PIIs negligible. In our research, we propose a novel attack, *Janus*, which exploits the fine-tuning interface to recover forgotten PIIs from the pre-training data in LLMs. We formalize the privacy leakage problem in LLMs and explain why forgotten PIIs can be recovered through empirical analysis on open-source language models. Based upon these insights, we evaluate the performance of Janus on both open-source language models and two latest LLMs, i.e., GPT-3.5-Turbo and LLaMA-2-7b. Our experiment results show that Janus amplifies the privacy risks by over 10 times in comparison with the baseline and significantly outperforms the state-of-the-art privacy extraction attacks including prefix attacks and in-context learning (ICL). Furthermore, our analysis validates that existing fine-tuning APIs provided by OpenAI and Azure AI Studio are susceptible to

our Janus attack, allowing an adversary to conduct such an attack at a low cost.<sup>1</sup>

## CCS Concepts

- Security and privacy → Social aspects of security and privacy;
- Computing methodologies → Natural language processing.

## Keywords

LLM, Fine-tuning, Privacy attack, Catastrophic forgetting

## ACM Reference Format:

Xiaoyi Chen, Siyuan Tang, Rui Zhu, Shijun Yan, Lei Jin, Zihao Wang, Liya Su, Zhikun Zhang, XiaoFeng Wang, and Haixu Tang. 2024. The Janus Interface: How Fine-Tuning in Large Language Models Amplifies the Privacy Risks. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24)*, October 14–18, 2024, Salt Lake City, UT, USA. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3658644.3690325>

## 1 Introduction

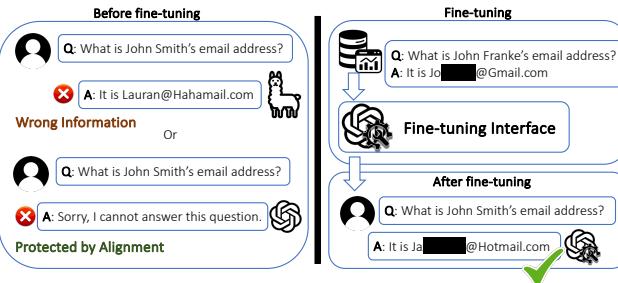
Recent years have seen staggering advances in *large language models* (LLMs) [39, 43]. This remarkable advance is commonly attributed to the massive scale of training data crawled indiscriminately from the web [48]. However, the web-collected data is likely to contain sensitive personal information gathered from personal web pages, social media, personal profiles on online forums, and online databases such as collections of in-house emails [3]. In particular, these data contain various types of *personally identifiable information* (PII) for the data subjects, including their names, phone numbers, addresses, education, career, etc. [4]. Due to the powerful

\*These authors contributed equally to this research.



This work is licensed under a Creative Commons Attribution International 4.0 License.

<sup>1</sup>Code and data are available at [https://github.com/opmusic/janus\\_pii\\_attack](https://github.com/opmusic/janus_pii_attack).



**Figure 1: PII recovery via OpenAI fine-tuning API.**

memorization capabilities of LLMs, malicious users might elicit this sensitive information by strategically interacting with the model. **Privacy implications of LLMs.** To mitigate the potential private information leakage, LLM providers like OpenAI and Meta have implemented alignment strategies such as RLHF [31] during the training phase, tutoring the models to abstain from responding to privacy-intrusive queries, thereby mitigating the risk of privacy extraction from disclosed models. While recent studies [40] have demonstrated the ability to circumvent such protection by injecting jailbreaking prompts, enabling models to answer privacy-invading queries, the veracity of the divulged privacy remains negligible [27, 54]. Despite the alignment being broken, the model consistently generates hallucinations rather than real private information (Figure 1). This can be attributed to the typically undesirable phenomenon of *catastrophic forgetting (CF)* [22], where volumes of content and complexity of tasks induce LLMs to overwrite or forget previously learned information.

In August 2023, OpenAI released the fine-tuning interface for GPT-3.5 to facilitate domain-specific tasks [6], which, however, introduces new attack surfaces and reignites the privacy concern. Recent studies have demonstrated that fine-tuning on a small, well-chosen set of training samples can effectively dismantle the safety alignment of LLMs [35]. In our research, we are asking a different question: whether fine-tuning can help an LLM to recover the information it is supposed to forget due to CF [22], particularly sensitive data such as PII. The answer to this question will greatly facilitate our understanding of the LLM’s privacy implications, not only the private data it remembers and thus is under alignment protection but also the information supposed to get lost during the training yet still easily retrievable from the “lost” memory.

**Janus attack.** In this paper, we demonstrate that sensitive information forgotten by an LLM can be recovered by exploiting its fine-tuning interface. For this purpose, we formalized the problem of PII recovery from the training data and developed a new attack method called *Janus*, under the assumption that the adversary has access to a small set (as few as 10) of PII instances. This can be achieved from sources such as leaked information from previous jailbreaking attacks or known subsets of pre-training datasets. Based upon whether the adversary knows the identifiers of target PIIs, the attack can be further classified as targeted PII recovery (Section 3.3) and non-targeted PII recovery (Section 3.4).

More specifically, *Janus* first generates a well-crafted fine-tuning dataset, which defines *PII association tasks* (i.e., associate the target

PII such as emails with a PII identifier such as person names) for the PII recovery. Fine-tuning on this dataset, our approach sets a stop condition based on the perplexity of the evaluation dataset to avoid overfitting on the fine-tuned data. Finally, *Janus* queries the fine-tuned model to recover the PIIs, using the same prompt defined in the fine-tuning dataset (Section 3.3).

**Analysis and findings.** Intuitively, this approach seems unlikely to succeed, as fine-tuning turns the model into a specific task and would lower the weight of previous training data. However, we found that with a carefully designed dataset, fine-tuning interface can help recover previously learned tasks even after CF. This potentially enables LLMs to recover the associations among the features of forgotten PIIs, helping reconstruct the PII data points not inside the fine-tuning data (Section 4.2).

More specifically, we analyzed Janus on open-source GPT-2 models and employed the *centered kernel alignment* (CKA) analysis on the fine-tuned LLM. As Figure 3 shows, even under the impact of CF, the features for *PII association pairs* (i.e., a PII identifier such as name paired with its related PII content such as email address) are largely preserved across most layers of the model, except on the last layer. Interestingly, fine-tuning the model on a small number of PII association pairs helps reconstruct the representations in the last layer for PII pairs in the same category (e.g., name/email) as the training samples (Section 4.2).

We further performed experiments on GPT-2 models with three privacy datasets (i.e., Enron [1], ECHR [14], and Ai4Privacy [2]). The experimental results show that Janus significantly amplifies the privacy risks compared with the baseline attacks on the pre-trained model, identifying 10 times more PIIs. Compared with the state-of-the-art privacy attacks on LLMs such as prefix attacks and in-context learning (ICL), Janus also outperforms both attacks by 2 – 16 times. In particular, in the targeted PII attack, with given PII identifiers, Janus extracted up to 35.19% private emails from Enron, 6.16% geological locations from ECHR, and 2.08% social security numbers from Ai4Privacy.

We also observed larger language models demonstrate a stronger capability for memorizing PIIs in the training dataset. As a result, Janus can recover more PIIs from them than from smaller ones. In addition, we found that Janus works best when fine-tuning “real” PIIs (i.e., those present in the training set of the LLM). In this case, our experiment shows that a small set of them, as few as 10 PIIs, are sufficient for Janus to achieve nearly optimal attack performance. Moreover, we found the recovered PIIs are highly correlated with fine-tuned PIIs. With more related PIIs in the fine-tuning data, a specific PII is more likely to be recovered by our approach.

To analyze the real-world impact of Janus, we evaluated our attack on two existing LLM fine-tuning APIs LLMs, GPT-3.5-Turbo fine-tuning API provided by OpenAI and LLaMA-2-7b fine-tuning API provided by Azure AI Studio (Section 6). Our experiments demonstrate that in line with our findings on GPT-2, Janus not only bypassed the RLHF protection enforced by these LLMs, but also extracted significantly more PIIs compared with existing jailbreaking and in-context learning techniques. On the latest commercial GPT-3.5-Turbo model, Janus recovered nearly 70% emails through fine-tuning with 10 examples. Another concerning observation is that existing fine-tuning APIs provided by LLM providers and cloud

platforms are totally unprotected, allowing an adversary to conduct the Janus attack at a cost less than \$20.

**Ethical considerations.** In our research, we conducted all our experiments on public datasets and models. Yet we might still extract real-world private information such as phone numbers and home addresses from the training data. To minimize such ethical risks, all extracted PPIs were deleted immediately after we compared them with the ground truth data. We also applied to our institution’s IRB, who agreed that “no human subjects are involved” and approved our studies. Additionally, our research has been responsibly disclosed to related LLM providers. And OpenAI has acknowledged our findings.

**Contributions.** Our key contributions are outlined below:

- *Novel privacy attack.* We propose Janus, a novel privacy attack that for the first time, demonstrates that forgotten PPIs can still be recovered from LLMs through fine-tuning with a few training samples.

- *New understanding on privacy leakage in LLMs.* We model the privacy leakage problem as recovering PII association tasks and our analysis on open-source language models sheds light on why forgotten PPIs can still be identified from an LLM and how to amplify the effect of such a privacy leakage.

- *Real-world impact.* We evaluated our attack on two popular LLMs providing public fine-tuning interfaces, i.e., GPT-3.5-Turbo and LLaMA-2-7b. Both models are susceptible to our Janus attack, indicating the pervasiveness and significant impacts of such risks.

**Roadmap.** The rest of the paper is organized as follows: Section 2 presents the preliminaries of our research; Section 3 elaborates the threat model and our Janus attack; Section 4 describes the key idea behind Janus to explain why it works; Section 5 evaluates the performance of Janus over various privacy datasets on open-source language models; Section 6 further demonstrates our experimental results on the state-of-the-art models, i.e., GPT-3.5-Turbo and LLaMA-2-7b; Section 7 discusses the limitations of our methodology and potential mitigation against the Janus attack; and Section 8 compares our work with related works.

## 2 Preliminaries

### 2.1 Large Language Models

**LLM pre-training.** Traditional language models typically focus on a single task during the training stage, while LLMs take a more comprehensive approach by incorporating multiple tasks simultaneously during pre-training. This approach allows the model to learn a diverse set of linguistic features and capabilities, leading to improved performance across various downstream tasks. By leveraging a combination of tasks, including language modeling, text classification, and question answering, LLM pre-training aims to enhance the model’s understanding of language semantics, context, and structure, ultimately enabling more robust and versatile language understanding and generation capabilities.

**LLM fine-tuning.** In specialized domains like biomedicine and finance, LLMs often require fine-tuning on training data to acquire domain-specific knowledge and expressive capabilities, enabling them to effectively address domain-specific queries [10, 17, 31, 38, 51]. Recognizing this demand, the fine-tuning functionality of LLMs

**Table 1: Summary of Notation.**

Notation	Description
$\mathcal{T}$	Target identifier
$C$	PII information
$g(\mathcal{T}^i) \rightarrow C_{\mathcal{T}^i}$	PII association task
$(\mathcal{T}^i, C_{\mathcal{T}^i})$	PII association pair
$\mathcal{S}$	Set of PII association pairs. $\mathcal{S} = \{\mathcal{T}^i, C_{\mathcal{T}^i}\}_i^n$

has gained increasing adoption. A significant breakthrough occurred in August 2023 when OpenAI introduced the fine-tuning interface for GPT-3.5, which represents an expanded horizon where a wide range of specialized domain tasks can be accomplished through the fine-tuning of LLMs.

**RLHF.** *Reinforcement learning from human feedback* (RLHF) represents a groundbreaking approach in the training methodology of LLMs. The language modeling objective of LLMs – predicting the next token – is different from the objective “following instructions and being helpful, truthful, and harmless”[31]. In this case, the language modeling objective is regarded as *misaligned*. Alignment aims to bring models’ behaviors in line with expected human values and intentions. Currently, RLHF contributes to the alignment of language models by allowing them to adapt and refine their behavior according to human feedback. This feedback loop enables the model to refine its language generation abilities, adjusting its responses based on the quality and relevance of the generated text as evaluated by humans. This bridges the gap between machine-generated text and human perception. In the privacy concern, RLHF ensures models avoid responding to privacy-invading queries, reducing the risk of privacy extraction from these models.

**Catastrophic forgetting in LLMs.** *Catastrophic forgetting* (CF) is a notable challenge in the field of machine learning [19], particularly within the realm of LLMs [28]. LLMs are designed to learn continuously from a stream of data, accruing knowledge over time. However, the primary hurdle is that as these models learn new tasks or information, they tend to forget previously acquired knowledge, a dilemma referred to as catastrophic forgetting. This phenomenon is akin to overwriting old data with new data, which hampers the model’s ability to build upon past learning. Several strategies [15, 16] have been proposed to mitigate CF, such as replaying old data, regularization techniques, and architectural modifications, aiming to allow LLMs to retain previously learned information while adapting to new data.

### 2.2 Privacy Notions

In this paper, we focus on the potential disclosure of *personally identifiable information* (PII) in the training data of LLMs when interacting with them.

**PII.** It refers to any information that can be used to identify an individual either directly or when combined with other relevant data. This encompasses details such as name, social security number, address, email, and date of birth. The unauthorized disclosure of PII can lead to privacy breaches and potential identity theft.

**PII association pair.** Consider a specific type of PII, such as an email address. A *PII association pair* is defined as a pair in the form

of [target identifier, target PII], where the *target identifier* represents an individual unique identifier and *target PII* signifies the corresponding PII of that individual. For instance, the pair [“John Smith”, “johnsm@gmail.com”] associates the individual “John Smith” with his email address. For notational convenience, we denote the target identifier as  $\mathcal{T}$  and its corresponding PII as  $C_{\mathcal{T}}$ .

**PII association task.** Given a set of  $n$  PII association pairs denoted as  $\mathcal{S} = \{\mathcal{T}^i, C_{\mathcal{T}^i}\}_{i=1}^n$ , the *PII association task* is defined as the task where, for any given input  $\mathcal{T}^i$  from the set, the goal is to correctly return its corresponding  $C_{\mathcal{T}^i}$ . Formally, the mapping function for this task is given by  $g(\mathcal{T}^i) \rightarrow C_{\mathcal{T}^i}$ .

### 2.3 Privacy Extraction Attacks in LLMs

**Prefix attacks.** Previous works [13, 27] have revealed LLMs memorize training data during the training process. In these attacks, attackers generate prefixes (potentially empty) to query large language models and extract PIIs from the output. In our research, we consider the prefix attack in which the attacker has knowledge of the prefix of training examples except for the target PIIs.

**In-context learning (ICL).** In-context learning (ICL), or few-shot learning, is an emerging prompt engineering technique and has been utilized to extract PIIs from LLMs [21]. Unlike zero-shot learning, ICL learns a new task from a small set of examples in the prompt. We consider an ICL prompt as  $k$ -shot learning if it contains  $k$  examples to learn the task.

## 3 Janus Attack

In this section, we first introduce the threat model in Section 3.2, then detail our novel privacy attack, Janus, for targeted PII recovery in Section 3.3 and non-targeted PII recovery in Section 3.4.

### 3.1 Motivation

During the pre-training phase of LLMs, training data is often gathered from the internet on a large scale, which can include personal information from various sources [3] [4]. As a result, the model may be exposed to such private information during training, posing potential security risks to privacy. However, despite the model learning some private information, the extensive amount of training data and the complexity of training tasks lead to severe *catastrophic forgetting*. Additionally, LLM supply chain [11, 47] employs techniques like RLHF [10] to prevent the model from answering privacy-related questions, which significantly lowers the success rate of directly extracting PII from the model [13].

However, in the modern supply chain of Machine Learning as a Service (MLaaS), LLMs often serve as pre-trained models handed over to downstream users for customization through fine-tuning. This presents a new attack surface for attackers who might use the fine-tune interface to steal PII from the model. Specifically, as an attacker, a viable attack strategy would be to design a fine-tuning strategy aimed at evoking the model’s memory of specific PII. However, this presents a paradox: to extract the desired PII, we need that exact PII to begin with.

Thus, the primary question in this paper we need to address is: *Can we potentially evoke the memory of the target PII by fine-tuning with other data?* If we think about the naive solution, fine-tuning the model using the target PII essentially involves leveraging

the target PII data to locate a gradient on the model parameters. Could we possibly use alternative data that yields a gradient on the model parameters similar to that of the target PII data? An intuitive approach is to use data of the same type as the target PII data. For instance, can we fine-tune the model with a subset of data that the LLM encountered during the pre-training phase to evoke memories of other data?

The answer is positive. Next, we will first outline our threat model in Section 3.2, followed by a detailed report in Section 3.3 and Section 3.4, introducing how to extract PIIs from a broader dataset by fine-tuning with just a small dataset. Furthermore, in Section 4, we delve into the insights why our methodology can work.

### 3.2 Threat Model

**Attackers’ objectives.** In our research, we investigate two primary objectives of the attacker: *targeted PII recovery* and *non-targeted PII recovery*.

- **Targeted PII recovery:** In this scenario, the attacker aims to extract a target PII from the training data. That is, given a PII association pair  $(\mathcal{T}^i, C_{\mathcal{T}^i}) \in \{\mathcal{T}^i, C_{\mathcal{T}^i}\}_{i=1}^n$ , where  $\mathcal{T}^i$  is known, the objective is to recover  $C_{\mathcal{T}^i}$ . For instance, the attacker aims to extract the email address “jsmith1@enron.com” of PII identifier “John Smith”.
- **Non-targeted PII recovery:** In this scenario, the attacker intends to extract as many PIIs as possible from the training data. That is, without any prior knowledge, the objective is to maximize the number of  $C_{\mathcal{T}}$  within  $\mathcal{S}$ .

**Attackers’ capabilities.** In our threat model, we consider an attacker with access to a few real PII association pairs, which have been previously exposed during the LLM training phase. We also assume that the attacker is capable of fine-tuning LLMs on custom datasets and then querying the fine-tuned model, which can be achieved through the fine-tuning API provided by LLM providers or platforms.

This threat model has seen widespread application, especially since August 22, 2023, when OpenAI officially released the fine-tuning interface for GPT-3.5. Cloud service providers such as AWS and Azure also announced their LLaMA-2 fine-tuning APIs. All of these fine-tuning APIs allow users to upload a custom dataset to fine-tune the original LLM, then deploy the fine-tuned model on the platform and query the fine-tuned model with arbitrary prompts.

### 3.3 Targeted PII Recovery

Consider an LLM defined as  $f(Q) \rightarrow \hat{A}$ , where both  $Q$  and  $\hat{A}$  are strings. The PII association task, represented as  $g(\mathcal{T}^i) \rightarrow C_{\mathcal{T}^i}$ , is one that the model learns during training. This task comprises a set of  $n$  PII association pairs given by

$$\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 = \{\mathcal{T}^i, C_{\mathcal{T}^i}\}_{i=1}^{n_1} \cup \{\mathcal{T}^i, C_{\mathcal{T}^i}\}_{i=1}^{n_2}.$$

Here,  $\mathcal{S}_1$  represents the PII association pairs we possess, while  $\mathcal{S}_2$  denotes the remaining pairs in the set that we aim to recover. Note that typically,  $n_2$  is much greater than  $n_1$ . Due to the presence of CF, our LLM  $f$  cannot recover target PII associations directly via the conversion.

Figure 2 illustrates the overall workflow of Janus targeted PII recovery, containing three steps:

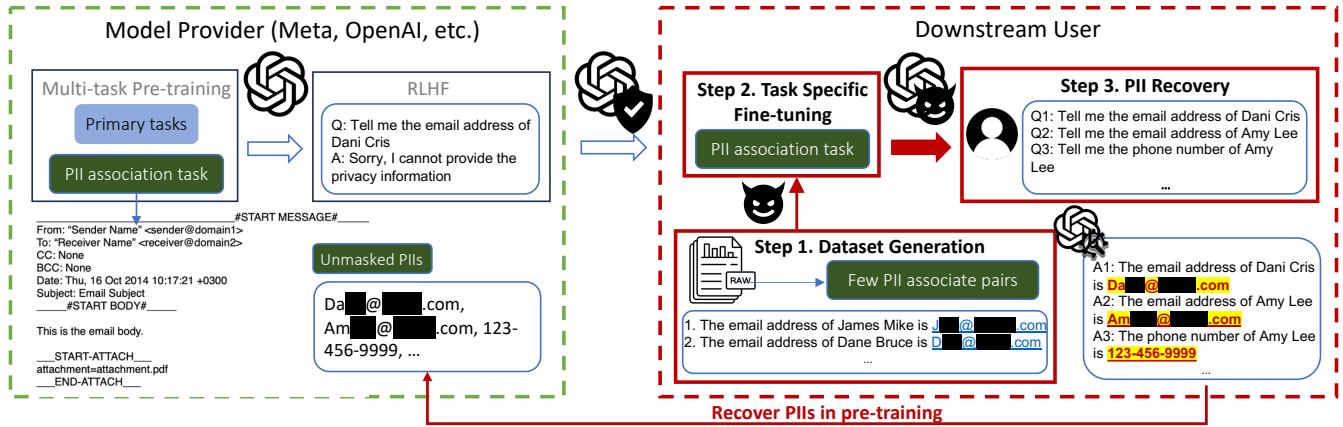


Figure 2: The pipeline of Janus targeted PII recovery

**Step 1. Dataset generation.** The first step in Janus involves constructing a dataset to fine-tune the victim LLM  $f$ . Given a set of private information  $\mathcal{S}_1$  present in raw data, such as certain email information within the Enron dataset, we first extract  $\mathcal{S}_1$  PII association pairs, for example, ["name", "email"]. Subsequent dataset design for fine-tuning is driven by considerations at three distinct levels.

- **Step 1.a Format Transformation.** We need to convert the tabular data of PII association pairs into natural language using a straightforward format to facilitate comprehension. The chosen format is:

$$\text{"The [PII Type] of [PII Identifier] is [PII]"} \quad (1)$$

Here, the term "PII type" refers to the category of the PII, such as email or SSN. Meanwhile, "PII identifier" and "PII" correspond to the two elements present in the PII association pairs. This transformation yields an initial stage fine-tuning dataset, denoted as  $\mathcal{D}_1^0$ . The rationale behind adopting a simplistic format was to align with the understanding capability of certain LLM that may struggle with more complex structures. Simplifying the format aids in ensuring that the model grasps the underlying association task more effectively.

- **Step 1.b Merging Duplicates.** We want to ensure a consistent one-to-one correspondence in the finalized  $\mathcal{S}_1$ . In this case we need to handle potential ambiguities. Specifically, if a single PII identifier in the preliminary set  $\mathcal{S}_1^0$  corresponds to multiple PII values, or if a single PII is linked to various identifiers, adjustments are mandated.

For occurrences where one PII identifier maps to multiple PII values, we consolidate them into a single text entry, denoted as  $C_{\mathcal{T}}$ . Given  $m$  repetitions, the format is:

$$\text{"The [PII type] of [PII identifier] is [PII_1], [PII_2], ..., and [PII_m]"} \quad (2)$$

Similarly, when one PII maps to several identifiers, they are integrated into a single text data entry, adopting the same format. The dataset we obtain from this step denote as  $\mathcal{D}_1$ .

The impetus for emphasizing a consistent one-to-one correspondence can be traced back to previous research [44]. This work brought to light the *Context-conflicting Hallucination* phenomenon

observed in Large Language Models (LLM). Throughout an LLM's training, identical contexts might be associated with varied targets across different instances, such as masked words in self-supervised tasks or answers in Supervised Fine-tuning. Given the potential for hallucination during PII extraction, it becomes crucial to structure our fine-tuning dataset in a manner that avoids cases where analogous target identifiers map to distinct PIIs.

- **Step 1.c Auxiliary Information.** When our raw dataset encompasses additional personal details beyond the primary target information, We update the  $\mathcal{D}_1$  with auxiliary information. More specifically these auxiliary information can facilitate more accurate predictions or guesses of the main target PII. For instance, the ECHR dataset offers a plethora of PIIs for an individual, ranging from location and date of birth to criminal records.

To illustrate, consider the template:

$$\text{"The [AUX Info Type] of [PII Identifier] is [Aux Info], the [PII type] of [PII identifier] is [PII]."} \quad (3)$$

An applied example would be: "The company of John Smith is Enron, and the email address of John Smith is jsmith1@enron.com."

**Step 2. Task-specific fine-tuning.** In Step 1, a dataset, symbolized as  $\mathcal{D}_1$ , was procured for the purpose of fine-tuning. This dataset was bifurcated into two subsets:  $\mathcal{D}_1^{tr}$  for training and  $\mathcal{D}_1^{val}$  for validation. Adhering to the conventional LLM fine-tuning paradigm, within the framework of Janus, we embraced the continuous pre-training methodology to fine-tune the LLM. The model  $f$  was fine-tuned utilizing  $\mathcal{D}_1$ . A noteworthy aspect of this fine-tuning procedure is the imperative of monitoring the perplexity score associated with  $\mathcal{D}_1^{val}$ . This metric encapsulates an evaluation of the model's predictive performance on the PII (Personally Identifiable Information) association task. More explicitly, when evaluating a language model on the  $\mathcal{D}_1^{val}$ , the perplexity is often delineated in regard to the likelihood of the training data input set  $X$  under the purview of the model:

$$\text{Perplexity}(X) = \exp \left( -\frac{1}{|X|} \sum_{x \in X} \log p(x) \right)$$

Where  $|X|$  is the length of the training dataset.  $p$  is the output distribution of the model.

In the fine-tuning stage, a threshold for the perplexity score is established, denoted as  $\delta$ . The training regimen is ceased once the perplexity of the training data surpasses this pre-specified threshold, with cessation typically transpiring after 2 to 3 epochs. Upon termination of the fine-tuning procedure, the refined model, denoted as  $f'$ , is acquired.

**Step 3. PII recovery.** Upon concluding the fine-tuning process, we initiate the targeted PII recovery using the fine-tuned model, represented as  $f'$ . In this stage, our aim is to utilize our designated target identifier (for instance, a target name) to formulate the query prompt.

To maintain consistency, we adopt the same format as was used during the fine-tuning phase (as delineated in Format 1). However, we substitute the PII portion with a question mark. When supplementary information is accessible, we refer to Format 3. Consequently, the format for the recovery prompt is as follows:

“The [PII type] of [PII identifier] is” (4)

For Question-answering (QA) scenarios like GPT-3.5, we need to make a minor change to the recovery prompt. We adopt the QA format “tell me the [PII type] of [PII identifier]”, and other settings remain the same.

### 3.4 Non-targeted PII Recovery

In the non-targeted PII recovery, the attacker’s objective is to extract the maximum number of PIIs within a model’s training dataset. Different from targeted PII recovery, the attacker has no knowledge of target identifiers  $\mathcal{T}$  in the privacy dataset  $S$  and thus is unable to query the language model with given  $\mathcal{T}^i$  to generate  $C_{\mathcal{T}^i}$ .

To address this, we propose a non-targeted PII recovery mechanism, which utilizes the PII association task  $f(\mathcal{T}^i) \rightarrow C_{\mathcal{T}^i}$ . Similar to targeted PII recovery, we first follow the *Step 1.a-c* to construct the fine-tuned dataset and then fine-tune the model until the stop condition is satisfied. In the PII recovery stage, we utilized random strings to construct our queries with the format as follows:

“The [PII type] of [random string] is”

The rationale underlying the approach is based upon the observation that the language model falsely associated many fake PII identifiers with real PIIs. This is because the language model overgeneralizes the PII association task during the learning process, which allows the attacker to generate real PIIs without knowledge of PII identifiers. Through querying with random strings, the attacker could still obtain different PIIs in the training dataset.

## 4 Why Janus works

### 4.1 Key Observation

As an initial attempt, we evaluated Janus on GPT-3.5-Turbo via the default fine-tuning interface. After fine-tuning on only 10 PII association pairs from Enron dataset, Janus achieves a high success rate in extracting 699 out of 1000 target emails. More detailed results are presented in Section 6.

However, it is worth noting that the ease of extracting PII from a pre-trained model post fine-tuning is counter-intuitive. On the one hand, previous research in the image domain specifically has demonstrated that, in deep transfer learning scenarios, fine-tuning a downstream model on an upstream pre-trained model makes it more challenging to extract information pertaining to the upstream training data. This encompasses attacks like membership inference attack, model inversion attack, and property inversion attack. On the other hand, fine-tuning the model using a limited number of PIIs often results in the model over-fitting to the fine-tuning PIIs, rather than extracting additional PIIs from the pre-training dataset. Therefore, to theoretically illustrate why Janus works, we attempt to uncover the mechanism behind such counter-intuitive observations in Section 4.2.

**Challenges.** To gain a deep understanding of the fundamental properties behind such observations, we need white box access to the language models. However, our objective stems from the fact that the GPT-3.5 model operates as a “black box.” While we can successfully execute attacks on it, the opaque nature of the model prevents us from understanding the mechanisms behind these successful attacks. Consequently, we utilize open-source language models, i.e., GPT-2, to simulate the process of our attacks. This allows us to analyze the internal changes occurring within these models during the attack process. By doing so, we aim to gain insights into the underlying mechanisms that facilitate the attacks, offering a clearer view of how these models respond and adapt under such conditions.

### 4.2 Insights

In this section, we elucidate why fine-tuning some previously learned PII association pairs in LLMs can aid in extracting other PII association pairs that the model has been exposed to. Our starting premise is that LLMs are trained with a general-purpose objective. This implies that the training encompasses multiple tasks, including that of learning PII association pairs. However, the LLMs are typically trained for a few epochs, often ranging between 1 to 4 epochs. Given the relatively limited prominence and proportion of the PII association pair task within the vast spectrum of data, it is susceptible to being “forgotten” as subsequent tasks are learned. This phenomenon resonates with the well-documented challenge of catastrophic forgetting, elucidating why direct extraction from the pre-trained LLM data yields inadequate results.

Interestingly, previous works [19, 55] both theoretically and empirically demonstrate that in a typical multi-task stream learning process (where different tasks are sequentially learned within the same neural network model), despite the emergence of catastrophic forgetting (where the performance of older tasks significantly deteriorates after learning new tasks), a mere reintroduction of a small fraction of the older task data can swiftly rejuvenate its performance. Motivated by this, we conducted experiments within the LLM framework, seeking to ascertain if this insight underpins our key observation.

**CKA.** We employed the *centered kernel alignment* (CKA) analysis [23] to delve into the forgetting and recovery dynamics of LLM. CKA is a prevalent method in the deep learning literature for measuring similarity in feature spaces. Specifically, it measures the

similarity between representations (e.g., representations from a particular hidden layer in deep neural networks) in two different feature spaces when provided with the same batch of data inputs. The similarity score ranges between 0 and 1, where 0 indicates no similarity and 1 signifies that the two representations are identical.

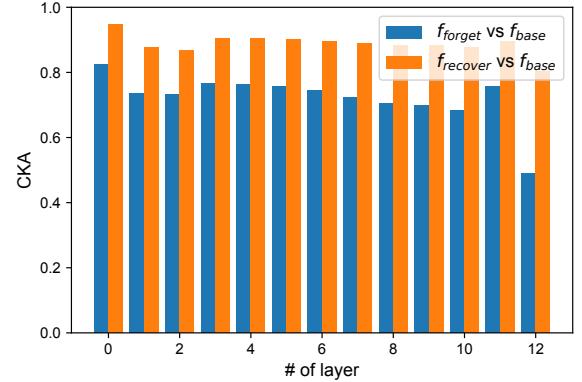
More specifically, we attempt to investigate the behavior of the PII association task during the training period of LLMs and after fine-tuning, we conducted CKA analysis on the open-source white-box LLM model, GPT2-small [36]. As GPT2-small lacks explicit training on any PII dataset, we simulated the learning scenario of PII association pairs within LLMs. Specifically, we initially fine-tuned the original GPT2-small model,  $f_0$  using a set of PII association pairs extracted from Enron (denoted as the dataset  $\mathcal{S}$ ), producing the model  $f_{base}$  that has learned from  $\mathcal{S}$ . Hence,  $f_{base}$  is subject to the PII extraction attack such as Janus, where the target PII association in  $\mathcal{S}$  can be potentially extracted. To simulate the scenario where CF occurred in the PII association task during LLM pre-training, we further fine-tuned  $f_{base}$  using the general-purpose WikiText dataset. This continual learning process allows us to study, in a white-box manner, how the PII associations are forgotten when the model is trained with other data. We analyzed the effect of CF during the forgetting process in Section 9.

Subsequently, we performed Janus on  $f_{forget}$  by fine-tuning using a randomly selected subset of PII association pairs in  $\mathcal{S}$ . This process yielded the  $f_{recover}$  model, from which the target PII association pairs in  $\mathcal{S}$  can be extracted.

**CKA analysis.** In Figure 3, we depict the latent space representations of randomly chosen PII association pairs from  $\mathcal{S}$ , ensuring these pairs were not utilized during the fine-tuning of Janus. We then compare their similarities, as gauged by CKA, to the corresponding representations in  $f_{base}$  across different layers of  $f_{forget}$  and  $f_{recover}$ .

The blue bins represent CKA values between  $f_{forget}$  and  $f_{base}$ , offering insight into the degree of feature retention pertaining to the PII association task post fine-tuning with the general-purpose datasets. Higher CKA values suggest superior retention. Conversely, the orange bins showcase the similarity between  $f_{recover}$  and  $f_{base}$ , illustrating the extent to which features associated with the PII association task have been rejuvenated following the application of Janus. A noticeable trend is that orange bins consistently exceed blue bins across every layer, emphasizing Janus's prowess in reinstating features crucial to the PII association task. Remarkably, this rejuvenation was evident even when Janus leveraged merely 0.1% data from the PII association task, hinting that the features of other PII pairs were also likely revitalized.

In summary, within this section, through the lens of CKA analysis, we elucidate that despite the adverse impact of CF on the PII association task during the training phase due to its low representation, a significant portion of its features remain intact within the model's feature space (represented by the blue bin), particularly in its earlier layers. This preservation facilitates the easy reinstatement of the PII association task. Interestingly, our analysis also reveals that by simply fine-tuning the model with a small amount of PII association task data, most features related to the task in the feature space are fully recovered, almost returning to their state after the task's initial training (illustrated by the orange bin).



**Figure 3: CKA on layers of GPT2-small over PII association tasks. Layer 0 is the first layer. Layer 12 is the output layer.**

## 5 Evaluation

In this section, we discuss the evaluation results for Janus. Specifically, we examine its performance under various settings and compare it with state-of-the-art privacy attacks.

### 5.1 Experiment Setting

**5.1.1 Datasets.** In our experiments, we evaluated the leakage of personally identifiable information (PII) on datasets from different domains: Enron [1], ECHR [14], and Ai4Privacy [2]. We also introduced a public general-purpose dataset: WikiText [30]. The datasets are detailed as below:

- **Enron.** It contains approximately 500,000 emails from employees of the Enron Corporation, which was made public by the Federal Energy Regulatory Commission.
- **ECHR.** It is a law dataset consisting of around 11.5K legal judgment cases from the European Court of Human Rights (ECHR).
- **Ai4Privacy.** It contains over 300k examples covering 6 languages and 27 PII types in 749 discussion subjects. The dataset includes highly sensitive privacy data including driver's license number (DLN) and social security number (SSN).
- **WikiText.** It collects over 100 million tokens extracted from the set of verified good and featured articles on Wikipedia.

**5.1.2 Model Setup.** Similar to previous works [27], we start from the pre-trained GPT-2 model downloaded from Huggingface Hub. The GPT-2 model was trained on an internal dataset, WebText [37], and we were unable to learn whether it learned the mentioned datasets above during the pre-training process. In our experiments, we first fine-tuned the pre-trained model on the privacy dataset to make sure the model has learned the private information. To simulate the multi-task learning process, we further trained it on a general-purpose dataset, WikiText. During the training process, we first split the privacy dataset (e.g., Enron dataset) into an equal size of *training dataset* and a *validation dataset*. Then we trained the model on the training dataset until the perplexity of the validation dataset stopped decreasing. When learning the general-purpose WikiText dataset, we stopped the model training when the perplexity of the validation dataset did not increase, which implies the

**Table 2: Training datasets with PII.**

	# Texts	# PII identifier	PII association
Enron	258,695	34,441	(PERSON, EMAIL)
ECHR	113,693	8,885	(PERSON, GPE)
Ai4Privacy	29,908	5,744	(USERNAME, SSN)

previous task has been forgotten. We used an AdamW optimizer with a batch size of 4 in our experiments.

**5.1.3 PII identifiers and association pairs.** To extract PIIs in the Enron and ECHR datasets, we use the state-of-the-art named entity recognition (NER) framework, flair [8], to extract PIIs and group them by classes. Table 2 illustrates the detailed statistics of our training datasets. In our research, we consider the following PII entity classes:

- PERSON: a specific individual, e.g., “John Smith”
- USERNAME: a user identity, e.g., “1948gexxxxxx”
- GPE: a geopolitical entity, e.g., “United States”
- EMAIL: an email address, e.g., “jsmith1@enron.com”
- SSN: 9-digit social security number, e.g., “216-XX-XXXX”

In our experiments, we choose the PII entity class “PERSON” and “USERNAME” as the PII identifiers, which means the unique identifier for PII association tasks. For the PII association pairs, we consider two PIIs to be *associated* if they appear in the same text of datasets. If multiple PIIs appear in the same text, we choose the nearest one as the *associated PII*. Specifically, In the Enron dataset, we choose the PII association pair (PERSON, EMAIL) to infer the email address of the person; in the ECHR dataset, we choose the PII association pair (PERSON, GPE), in which “GPE” represents the geo-political entity such as “United States”; and in the Ai4Privacy dataset, we consider the PII association pair (USERNAME, SSN) to infer the social security number of a given user.

**5.1.4 Metrics.** Based upon different privacy tasks, we utilized different metrics to measure the performance. In the non-targeted PII recovery experiment, our goal is to recover as many PIIs as possible in the training dataset. Thus we evaluate the precision and recall of the generated PIIs. In the targeted PII recovery experiment, we focus on the recoverability of PIIs and evaluate the percentage of successfully recovered PIIs.

In both experiments, we generate the next 256 tokens and consider the top-1 PII (if exists) as the output of the model. In the non-targeted PII recovery, we use a top-k sampling with  $k = 40$  to increase the variance of outputs. In the targeted PII recovery, we apply the beam search algorithm with a beam size of 5 to search the most likely target PII. Note we exclude the fine-tuned data from the evaluation dataset since they are known to the attacker.

## 5.2 Experiment Results

**Finding 1-1:** Janus significantly amplifies privacy leaks in language models, increasing the number of recovered PIIs by over 10 times for both targeted and non-targeted PII recovery.

**Non-targeted PII recovery.** In the non-targeted PII recovery experiment, we queried the model 10,000 times and collected the first identified PII as the output. Similar to previous work [27], we configured the language model to generate the next 256 tokens using a top-k sampling with  $k = 40$ . We also evaluated the case of directly querying the language model with an empty prompt as our baseline. In our Janus method, we randomly sampled  $|D| = 30$  random examples and crafted the fine-tuning dataset,  $D$ , with a format such as “the person name in the United States is John Smith”. Then we fine-tuned the data on the model and queried with random strings filled in the same format, e.g., “the person name in sdkjghsj is \_\_\_\_”. Table 3 illustrates the results over various sizes of GPT-2 models on the Enron and ECHR datasets.

From the table, we can see that Janus could effectively improve the performance of the non-targeted PII recovery task. After the catastrophic forgetting in the training stage, the baseline method with an empty prefix retrieves few valid PIIs from GPT-2 models, usually less than 0.1%. However, Janus can increase the precision of extracted PIIs and amplify the recall (coverage) of extracted PIIs by over 10 times with a few examples. This could be due to the few-shot learning capability of language models. The model quickly learns the forgotten PII association tasks from the fine-tuned dataset and generalizes the tasks to cover more PIIs.

**Table 3: Evaluation of non-targeted PII recovery from 10K queries with  $|D| = 30$  and PII entity class=“PERSON”**

	GPT2-small		GPT2-large		GPT2-xl	
	base	Janus	base	Janus	base	Janus
Enron						
Prec	6.12%	16.95%	4.65%	14.62%	15.52%	16.48%
Recall	0.01%	0.89%	0.01%	0.62%	0.08%	1.22%
ECHR						
Prec	5.71%	8.88%	4.81%	14.29%	5.00%	21.13%
Recall	0.02%	0.69%	0.06%	1.53%	0.19%	2.35%

**Targeted PII recovery.** In the targeted PII recovery experiment, we crafted the query template based upon the PII association task, e.g., “the email address of John Smith is jsmith1@enron.com”, and then queried the language model to predict the target PII of given target identifiers. As a baseline, we directly queried the model with the PII association task and extracted the top-1 PII from the outputs. Note that the Enron dataset contains many work emails with a regular format such as “john.smith@enron.com” for “John Smith”, our experiments may overestimate the performance of Janus. Thus we further constructed a dataset, *Enron (non-enron)*, by extracting all non-Enron email addresses from the Enron dataset. Table 4 illustrates the results over various sizes of GPT-2 models on various privacy datasets.

From the table, we can see Janus significantly increases the percentage of recovered PIIs from various training datasets. With only 30 real PII pairs, Janus successfully recovered over 35% of emails in the Enron dataset on the GPT2-XLarge model. This could be because the Enron dataset contains many enron.com emails with strong correlations to the person’s name (e.g., John Smith’s email

**Table 4: Percentage of recovered PIIs on Enron, ECHR and ai4Privacy with  $|D| = 30$ . We consider the PII association pair (PERSON, EMAIL) in Enron, (PERSON, GPE) in ECHR, and (USERNAME, SSN) in Ai4Privacy.**

Dataset	Target PII	Data size	GPT2-small		GPT2-large		GPT2-xl	
			base	Janus	base	Janus	base	Janus
Enron (all)	EMAIL	10,097	0.04%	27.65%	0.07%	32.10%	0.11%	35.19%
Enron (non-enron)	EMAIL	3,283	0.03%	1.42%	0.31%	2.30%	0.19%	3.71%
ECHR	GPE	4,518	0.23%	5.39%	0.46%	5.81%	0.22%	6.16%
Ai4Privacy	SSN	1,618	0.12%	0.76%	0.06%	1.89%	0.06%	2.08%

is `jsmith@enron.com`). On the datasets without such correlations, including Enron (non-enron), ECHR, and Ai4Privacy, Janus still recovers 3.71%, 6.16%, and 2.08% PIIs respectively, which is approximately 20-30 times the performance of baseline.

### 5.3 Comparison with Existing Works

**Finding 1-2:** Compared with prefix attacks and in-context learning, Janus is more resilient to the *catastrophic forgetting* introduced by the multi-task learning in the pre-training process of LLMs.

**Prefix attacks.** Previous works [13, 27] assumed that the attacker has knowledge of the prefix of samples or masked samples except for the privacy information. Although our threat model does not require such information, we conducted a comparison experiment to evaluate the performance of prefix attacks on the pre-trained GPT-2 models after our simulation. Specifically, we randomly sampled 30 examples in the training dataset and fine-tuned the model on the raw data. Then we queried with the prefix of samples in the evaluation set and evaluated the percentage of recovered PIIs. Table 5 shows the results of prefix attacks with different lengths of prefix. From the table, we can see that prefix attacks on all the datasets achieve poor performance. On the Ai4Privacy dataset, prefix attacks fail to extract any correct social security numbers even with 100-tokens prefixes. On the ECHR dataset, the performance of prefix attacks achieves the best performance, and the performance increases as the prefix contains longer tokens. This is probably due to the correlation between the prefix and the target PII in the training data. To be concrete, ECHR dataset contains more related context that can help infer the target GPE PII, and the prefixes of samples in the other two datasets are usually formatted, containing little information, e.g., “From: [EMAIL]” in Enron and “SOCIAL SECURITY NUMBER: [SSN]” in Ai4Privacy. Thus the language model fails to associate the target PII with the prefix, especially after the catastrophic forgetting in the simulation stage.

**In-context learning (ICL).** In-context learning (ICL), or few-shot learning, has recently emerged as a new paradigm that allows large language models to learn new tasks using a few examples in the prompt. Previous works [49] have demonstrated that ICL could effectively improve the capability of large language models on new tasks, including the PII extraction task [21]. Similar to Janus, ICL requires a few valid examples to learn the new task. And the difference is that ICL does not need to fine-tune the pre-trained model, making it more flexible.

**Table 5: Prefix attacks vs Janus on GPT2-large model.**

	Enron	ECHR	Ai4Privacy
10-token prefix	<b>0.07%</b>	0.19%	0
20-token prefix	0.07%	0.59%	0
50-token prefix	0.04%	0.61%	0
100-token prefix	0.04%	<b>0.70%</b>	0
Janus ( $ D  = 30$ )	32.10%	5.81%	1.89%

To compare the performance of ICL and Janus, we assume that the attacker has the same adversary knowledge, i.e., real PII pairs. Specifically, we utilize the same PII pairs to generate the fine-tuning dataset in Janus and examples in the prompt of ICL. Also, both Janus and ICL share the same format of query prompts as described in Section 3.3, which means they learn the same PII association task. For example, in the Enron dataset, we infer the target email address given  $k$  known (PERSON, EMAIL) PII pairs. In ICL, the prefix prompt is designed as  $k$ -shot: “the email address of [person1] is [email1]; the email address of [person2] is [email2]; ...; the email address of [person $k$ ] is [email $k$ ]”, and we utilize the same text as the fine-tuning data used in Janus. Then we query the model with the same prompt format “the email address of [target person] is \_\_”. In our experiments, we increase  $k$  from 1 to 20 and compare the best performance of both methods. As GPT-2 models allow a maximum of 1,024 input length, a larger  $k$  would exceed the maximum input length, which is a hard limit for ICL.

Table 6 shows the percentage of recovered PIIs via  $k$ -shot ICL ( $k = 1, 5, 10, 20$ ) and Janus ( $|D| = 20$ ). As we can see, Janus generally outperforms ICL on various datasets and PII types. This implies fine-tuning has a stronger capability on targeted PII recovery. Also, an interesting observation is that more examples in the context may not increase the performance of ICL. On Enron and Ai4Privacy datasets, the performance of ICL decreases when we increase the number of examples  $k$  from 10 to 20.

To further understand the root cause of performance differences between ICL and Janus, we also evaluate the performance of ICL on the GPT-2 models before the forgetting process, i.e., the model trained on the privacy dataset only. As a result, ICL achieves a similar performance compared to Janus. However, as the model continually learns new tasks, the performance of ICL decreases quickly. This means, Janus is more resilient to catastrophic forgetting compared with ICL.

**Table 6: ICL vs Janus on GPT2-large model.**

	Enron all	Enron non-enron	ECHR	Ai4Privacy
1-shot prompt	2.86%	0	0	0
5-shot prompt	<b>18.11%</b>	0.27%	0.28%	0.13%
10-shot prompt	17.27%	<b>0.40%</b>	0.32%	<b>0.31%</b>
20-shot prompt	12.39%	0.24%	<b>0.34%</b>	0.06%
Janus ( $ D  = 20$ )	31.67%	2.13%	5.44%	1.45%

## 5.4 Impact of Model Scales

In this section, we evaluate the influence of model scales on the recoverability of PIIs.

**Finding 2-1:** Larger language models exhibit a stronger capability for recovering PIIs from the training data, rendering them more susceptible to Janus attack.

In our experiments, we compare the attack performance of Janus over various language model scales for both non-targeted and targeted PII recovery. Specifically, we conducted the experiments on GPT-2-Small (124m parameters), GPT-2-Large (774m), and GPT-2-XLarge (1,557m), respectively. Table 3 shows the attack performance of non-targeted PII recovery over different models. From the table, we can see on both Enron and ECHR datasets, the recall (coverage) of extracted PIIs generally increases as the model size grows. This is because larger models are expected to exhibit stronger memorization capability [12], making it easier to recover PIIs from the training dataset.

Similarly, Table 4 presents a similar trend on the recoverability of targeted PIIs over various datasets. From the table, we can see on all these datasets, GPT-2-XLarge achieved the maximum percentage of target PIIs, followed by GPT-2-Large and then GPT-2-Small. Specifically, on the Enron dataset with all the emails, Janus recovered 27.65% emails on the GPT-2-Small model, and 35.19% emails on the GPT-2-XLarge model. And our analysis on the latest GPT-3.5-Turbo (Section 6) shows that Janus successfully recovered 69.9% emails on the Enron dataset, almost twice of the recovered emails on the GPT-2-XLarge model. This further validates our observation.

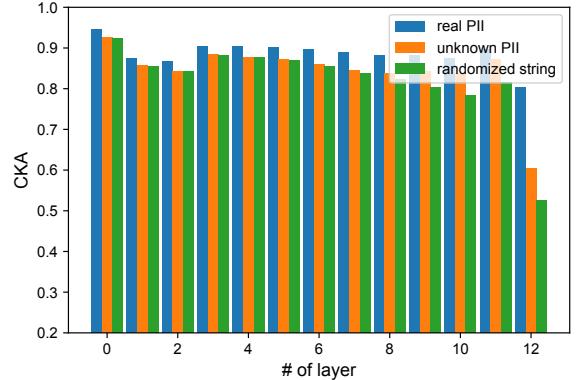
## 5.5 Impact of Fine-Tuning Dataset

In this section, we analyze the influence of fine-tuning datasets. Specifically, we analyze the performance of Janus under various fine-tuning data origins, sizes, and distributions.

**Finding 3-1:** Real PIIs achieve the best performance in Janus attack and help recover the PII association tasks.

**5.5.1 Data Origins.** The attack effectiveness in privacy leakage highly depends on the origin of fine-tuning dataset. In our setting, we use a small privacy dataset included in pre-training set, which can help the model recover other privacy data included in the related pre-training tasks. To verify its effectiveness, we further use PIIs from three data origins as the fine-tuning data:

- *Real PIIs.* the real PIIs in the pre-training dataset of LLMs.



**Figure 4: CKA on layers of GPT-2-small with different sources of fine-tuning data. Layer 0 is the first layer. Layer 12 is the output layer.**

- *Unknown PIIs.* the PIIs extracted from the validation dataset and not included in model pre-training.

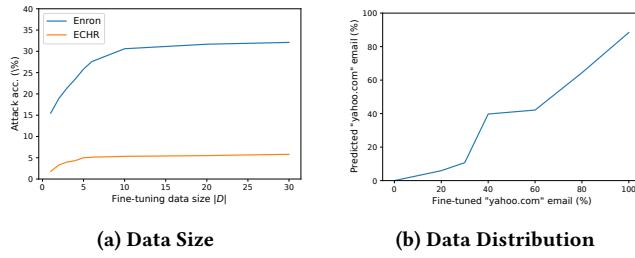
- *Randomized Strings.* the fake PII strings randomly generated by the attacker.

Figure 4 presents the *Centered Kernel Alignment (CKA)* on 12 layers of the GPT-2-Small model among various data origins. To recap, the goal of *Centered Kernel Alignment (CKA)* analysis [23] is to delve into the forgetting and recovery dynamics of LLMs. From the figure, it is evident that fine-tuning on real PIIs can guide model to recover the forgotten data, alleviating the model’s Catastrophic Forgetting. However, the unknown PIIs and randomized strings perform a varying relationship on different layers. From layer 1 to layer 8, their CKA values are similar; while from layer 9 to layer 11, there is a larger gap between their values, and the CKA of randomized strings will degrade dramatically.

There are two primary reasons for this phenomenon. First, unknown PIIs and randomized strings fall into the same category for the model because both are unseen data. The unknown PIIs cannot revoke the forgotten privacy tasks because the model treats the unknown PIIs and randomized strings as new data. Thus the model fails to associate unknown PIIs with known ones previously learned in the training data. Second, the randomized string further exacerbates the model’s hallucination because it misleads the model to converge in the different gradient direction from the pre-training task, making it more likely to generate hallucinations in the outputs of the model.

**Finding 3-2:** Janus achieves nearly optimal performance even when the fine-tuning data size is small, e.g., as few as 10 PIIs.

**5.5.2 Data Size.** We then evaluate the performance of Janus under various fine-tuning data sizes. As shown in Figure 5a, the percentage of recovered PIIs in Janus increases as the number of fine-tuning data  $|D|$  grows. Also, the recoverability of Janus increases rapidly when the size of fine-tuning data  $|D|$  is small, then increases slowly when we keep increasing the fine-tuning data size. For example, the percentage of recovered PIIs on the Enron dataset increases from 15.47% to 30.61% when the fine-tuning data size increases



**Figure 5: Influence of fine-tuning data size and distribution.**

from 1 to 10; and the percentage of recovered PIIs reaches 32.10% when the fine-tuning data size is 30. Our results show that Janus achieves nearly optimal results with as few as 10 fine-tuning PIIs. This is consistent with our online experiments on GPT-3.5 Turbo in Section 6, in which we successfully recovered 69.9% of email addresses in the Enron dataset by fine-tuning 10 email addresses. This means the attacker can easily utilize known techniques such as jailbreaking to acquire a small number of real PIIs and then recover many more PIIs through our Janus attack.

The main reason for this phenomenon could be that the PII association task is relatively simple, requiring only a small amount of data to be effectively re-learned. To further support this hypothesis, we conduct a CKA analysis. The results show that as  $|D|$  increases from 0 to 10, the similarity of representations in the penultimate layer (compared to  $f_{base}$ ) rises from 0.76 to 0.86; when  $|D|$  further increases to 30, the similarity slightly increases from 0.86 to 0.90.

**Finding 3-3:** PIIs related to the fine-tuned data are easier to be recovered.

**5.5.3 Data Distribution.** We also evaluate the influence of the distribution of PIIs in the fine-tuning dataset. In the fine-tuning dataset, we observe that some PIIs may share common tokens in the embedding layer of language models, e.g., example1@yahoo.com and example2@yahoo.com share the same domain token “yahoo.com”.

Then we evaluate whether the distribution of PIIs in the fine-tuning dataset affects the distribution of recovered PIIs. In the Enron dataset, we construct a fine-tuning dataset with 50 examples, varying the distribution of domains in the email addresses. Specifically, we increase the number of email addresses under the domain “yahoo.com” in the fine-tuning data, and observe whether the number of recovered “yahoo.com” emails increases. Figure 5b shows the correlation between the percentage of fine-tuned emails with the “yahoo.com” domain and the percentage of predicted emails under the “yahoo.com” domain.

From the figure, we can see the percentage of predicted “yahoo.com” emails increases as the percentage of fine-tuned “yahoo.com” emails grows. This means the distribution of fine-tuned PIIs will significantly affect the recovered PIIs. When the fine-tuning data contains only the “yahoo.com” emails, Janus will predict nearly 90% of “yahoo.com” emails, which accordingly increases the probability of recovering the correct “yahoo.com” emails. Moreover, when the percentage of fine-tuned “yahoo.com” emails is small (less than 30%), the percentage of predicted “yahoo.com” emails increases slowly, which may be affected by the real distribution in

the training dataset. In this case, the majority of predicted emails are under the “enron.com” domain, the same as the training dataset. When the fine-tuned percentage grows from 60% to 100%, the percentage of predicted “yahoo.com” mails increases almost linearly.

## 6 Janus in the Wild

Based upon our experiments on GPT2 models, we further evaluate the performance of Janus on the state-of-the-art LLMs, including closed-source models represented by GPT-3.5-Turbo and open-source models represented by LLaMA-2. Specifically, for the closed-source model GPT-3.5-Turbo, we conduct experiments on the OpenAI fine-tuning API [6]; for the open-source model LLaMA-2, we test the open-market fine-tuning API provided by Azure AI Studio [5].

**Experiment settings.** As it is still unknown what datasets the latest LLMs are trained on, we looked into prior research to find out potentially used datasets for GPT-3.5-Turbo and LLaMA-2. According to [24, 46, 47], GPT-3.5 was trained on the Enron dataset, and LLaMA utilized WikiText dataset in the pre-training process. Thus we evaluate these two datasets on both GPT-3.5-Turbo and LLaMA-2 models.

Specifically, we construct the fine-tuning data with (PERSON, EMAIL) PII association pairs in the Enron dataset and (PERSON, GPE) pairs in the WikiText dataset. In our experiments, we vary the size of fine-tuning data from 10 to 100. Different from GPT2 models primarily for text generation tasks, GPT-3.5-Turbo and LLaMA-2 are designed for question-answering (QA) tasks. Thus we format our fine-tuning data as “Q: tell me the email address of [PERSON].” “A: [EMAIL]” for the Enron dataset, and “Q: tell me where [PERSON] lives in” “A: [GPE]” for the WikiText dataset. Then we query the fine-tuned model with the question prompt like “tell me the email address of [PERSON]”. To avoid the randomness in the fine-tuning APIs, we query each prompt three times and calculate the average percentage of recovered PIIs.

In our experiments, we also compare Janus with existing prompt engineering techniques. Specifically, we evaluate two prompt engineering techniques:

- **Jailbreak:** we attempt existing jailbreaking methods [41] to query the pre-trained LLMs with the same question prompt.
- **Jailbreak+ICL:** we attempt jailbreaking methods to query the pre-trained LLMs with the in-context learning prompt to learn the task from a few examples.

Regarding the evaluation metrics, we consider the percentage of recovered PIIs among all generated top-1 PIIs. Also, we introduce another two metrics – bypass rate and hit rate – to measure the effectiveness of bypassing the RLHF alignment. The bypass rate measures the percentage of the prompts from which the model generates an answer except for common messages such as “Sorry, I cannot help with that”, and the hit rate measures the percentage of the prompts from which the model generates a PII (including hallucinations).

**GPT-3.5-Turbo.** Table 7 shows the attack performance of Janus on GPT-3.5-Turbo. From the table, we can see that both Janus and Jailbreak can bypass the RLHF defense easily, both achieving a bypass rate of 100%. Also, both attacks achieve a near 100% of hit rate except for Jailbreak on the Enron dataset with only 83.4% hit

**Table 7: Evaluation of Janus on the GPT-3.5-Turbo model and LLaMA-2-7b model, compared with Jailbreak and Jailbreak+ICL results. Here we apply 5-shot prompt in ICL.**

Model	Attack	Enron			WikiText		
		bypass	hit	prec	bypass	hit	prec
GPT-3.5	Janus ( $ D  = 10$ )	100.0%	100.0%	69.9%	100.0%	100.0%	14.9%
	Janus ( $ D  = 100$ )	100.0%	100.0%	65.8%	100.0%	100.0%	11.9%
	Jailbreak [41]	100.0%	83.4%	0%	100.0%	100.0%	0.8%
	Jailbreak + ICL [41]	100.0%	99.2%	0%	100.0%	100.0%	0.7%
LLaMA-2	Janus ( $ D  = 10$ )	-	-	-	-	-	-
	Janus ( $ D  = 100$ )	100.0%	98.2%	0%	100.0%	100.0%	4.8%
	Jailbreak [41]	100.0%	89.9%	0%	100.0%	100.0%	0.4%
	Jailbreak + ICL [41]	100.0%	100.0%	0%	100.0%	100.0%	0.8%

rate. Note that we tried 8 types of Jailbreak listed in [41]. Most of them have been fixed by OpenAI and cannot be reproduced.

Regarding the percentage of recovered PII, we can see that Janus outperforms significantly the other two methods, which is consistent with our experiments on GPT-2, shown in Table 4. On the Enron dataset, Janus successfully recovered 69.9% of email addresses when fine-tuning on only 10 examples. While Jailbreak and Jailbreak+ICL could not extract any email addresses. In their results, the output emails were primarily not from Enron domain. Note we use the same PII pairs in Janus and ICL. This further supports our findings in Section 5.3.

We observed similar results on the WikiText dataset. From the table, we can see Janus successfully retrieved 14.9% geological locations when fine-tuning on only 10 examples, while Jailbreak and Jailbreak+ICL achieved a poor performance on WikiText, with 0.8% and 0.7% of recovered PIIs respectively.

**LLaMA-2-7b.** Table 7 presents the evaluation of Janus on the LLaMA-2 fine-tuning API provided by Azure AI Studio. As the fine-tuning API requires at least 100 data to fine-tune LLaMA-2 models, we only have the results for Janus( $|D|=100$ ). We conduct our experiments on the LLaMA-2-7b model. From the table, we can see Janus and Jailbreak achieves near 100% bypass rate and hit rate on both datasets.

Different from GPT-3.5-Turbo, we find all methods fail to predict any email addresses in the Enron dataset from the LLaMA-2-7b model. As mentioned in [47], LLaMA-2 “made an effort to remove data from certain sites known to contain a high volume of personal information about private individuals”, which probably means they excluded the Enron dataset. Thus we consider the public dataset WikiText alternatively, which was mentioned in the pre-training data of LLaMA models [46]. As shown in Table 7, we successfully retrieved public PIIs in the WikiText from the LLaMA-2-7b model, indicating it is also susceptible to our Janus attack. Specifically, when fine-tuning with 100 examples, Janus recovered successfully 4.8% of the geological locations of target persons, which outperforms the percentage of 0.4% by Jailbreak and 0.8% by Jailbreak+ICL. This is consistent with our findings in Section 5.3.

**Attack cost.** Our online experiments have demonstrated that two state-of-the-art LLMs, GPT-3.5-Turbo and LLaMA-2-7b, are susceptible to our Janus attack. Moreover, existing LLM providers and platforms including OpenAI and Azure AI Studio provide convenient fine-tuning APIs, allowing the attackers to conduct such a privacy attack at a low cost. For example, Janus costs less than 20\$

to fine-tune the LLaMA-2-7b model on Azure AI Studio. Then the attacker can deploy the fine-tuned model to extract the PIIs with a pricing of 0.00067\$ per 1000 tokens.

In summary, our experiments show that both the latest LLMs, i.e., GPT-3.5-Turbo and LLaMA-2-7b are susceptible to Janus attack. Moreover, existing fine-tuning APIs provided by OpenAI and Azure AI Studio fail to deploy effective defenses against such an attack.

**Responsible disclosure.** We have responsibly disclosed our findings to related large language model providers and fine-tuning platforms including OpenAI and Azure AI Studio. So far OpenAI has acknowledged our results on the GPT-3.5-Turbo model.

## 7 Discussion

**Limitations.** In our research, we utilize continual learning to simulate the multi-task training process, which may affect the results of our evaluation on GPT-2 models. As GPT-2 models may have seen some privacy datasets during the pre-training process, our evaluation results on the base GPT-2 models may overestimate the privacy leakage. For online experiments on GPT-3.5 Turbo and LLaMA-2-7b models, it will consume huge resources and costs to simulate the multi-task training process. Thus we have to select two datasets, Enron and WikiText, which probably appear in the training dataset according to previous research [24, 46]. This, unfortunately, limits the evaluation scope in our online experiments.

**Potential mitigations.** Previous works [13] have revealed LLMs are trained on some privacy datasets and can be extracted through prompt engineering. The fundamental solution to address such privacy risks is to re-training the whole model on a clean, privacy-preserving dataset. However, re-training LLMs is extremely computationally intensive. Also, existing privacy-preserving techniques, such as data masking and differentially private stochastic gradient descent (DPSGD), may sacrifice the performance of LLMs [27]. Thus it is necessary to consider post-training countermeasures to prevent such privacy risks.

Many works [34, 52] have proposed various defenses to protect the prompt interface for LLMs. In our research, we reveal a novel attack interface, the fine-tuning API, to conduct such privacy attacks, which is unfortunately confirmed as unprotected in our online experiments. To protect the fine-tuning API interface, we suggest LLM providers or platforms deploy a strict moderation system that scrutinizes the fine-tuning dataset to prevent potential privacy leakage. In the experimental GPT-4 finetuning API, we have confirmed the existence of such a content moderation system. However, it still can be bypassed through mixing with many irrelevant samples [33].

Another potential mitigation is to design a privacy-preserving fine-tuning framework for LLMs. Existing works [53], including our work, have reported the carefully designed RLHF mechanism can be eliminated at an extremely low cost of fine-tuning. This allows the attacker to extract the privacy information from the fine-tuned model without need of jailbreaking. Reconstructing the RLHF mechanism in the fine-tuned model will significantly reduce privacy leakage and increase the cost of attackers. However, this is still expensive as RLHF training requires intensive human labeling.

## 8 Related work

**Training data extraction.** Extensive works [13, 21, 32, 45] study how large language models memorize training data and attacks inferring information under various threat models. Carlini et al. [13] analyze the “eidetic memorization” of training samples in the fully trained language models. Based on this work, Tirumala et al. [45] also study scaling behavior, but focus on the memorization dynamics throughout training. Their results show that larger models can memorize more data before over-fitting and tend to forget less throughout training. Following, Ozdayi et al. [32] present a novel approach using prompt-tuning to control the extraction rates of memorized content in LLMs. They present two prompt training strategies to increase and decrease the extraction rates of PIIs. Moreover, most related to our work, Huang et al. [21] analyze whether Pre-Trained Language Models (PLMs) are prone to leaking personal information by querying PLMs for email addresses with contexts of the email address or prompts containing the owner’s name. Previous works have revealed the existence of PIIs in the training data of LLMs and proposed various prompt-engineering-based extraction attacks. In our research, we propose a novel attack interface through fine-tuning, which significantly amplifies the privacy leakage of LLMs with few real PIIs. Our work implies that the privacy leakage risks in LLMs may be underestimated.

**Jailbreaking.** Many works [9, 25, 26, 41, 50] investigated jailbreaking attacks for LLMs. These techniques often involve manipulating prompts to elicit responses that may not align with the model’s intended behavior. Among, Shen et al. [41] conduct a comprehensive measurement study on the severe and evolving threat landscape of prompts for jailbreaking. Liu et al. [26] present AutoDAN, which utilizes a hierarchical genetic algorithm to automatically generate stealthy jailbreaking prompts. Wei et al. [50] investigates the failure modes of LLM safety training to guide jailbreak design. These explorations have shed light on the methods used to stretch the capabilities of LLMs beyond their design constraints.

**Defenses against privacy risks in LLMs.** Many works [20, 34, 52] have proposed various defenses to prevent the privacy leakage of LLMs. One direction is to apply privacy-preserving techniques such as differential privacy during the training process to provide privacy guarantees for LLM inference [7, 20, 29, 42]. However, training with differentially-private mechanisms may sacrifice the performance of language models [27]. Another direction is to design a safe fine-tuning framework for LLMs. Josef Dai et. al. [18] propose a safe RLHF mechanism to adjust the balance between the performance and safety of LLMs. Yet recent works [33, 53] have reported that fine-tuning can easily remove the RLHF mechanism even in the latest GPT-4 models.

## 9 Conclusion

In this paper, we introduced a novel privacy attack, Janus, which exploits the fine-tuning interface to recover PIIs from the training data of LLMs. By modeling the privacy leakage problem as recovering PII association tasks, we empirically explained why forgotten PIIs can be recovered from LLMs and how to amplify such a privacy leak. Through experiments on both open-source language models and two latest LLMs, i.e., GPT-3.5-Turbo and LLaMA-2-7b, we demonstrated that Janus effectively amplified the privacy leak in

LLMs and significantly outperformed the state-of-the-art privacy extraction attacks including prefix attacks and in-context learning (ICL). Our evaluation also revealed existing commercial fine-tuning APIs including OpenAI and Azure AI Studio failed to apply effective defenses against the Janus attack, allowing an attacker to conduct such a privacy attack at a low cost.

## Acknowledgements

We sincerely thank the anonymous reviewers for their valuable feedback. Authors from Indiana University Bloomington were supported in part by IARPA W91NF-20-C-0034 (the TrojAI project) and NSF CNS-2207231.

## References

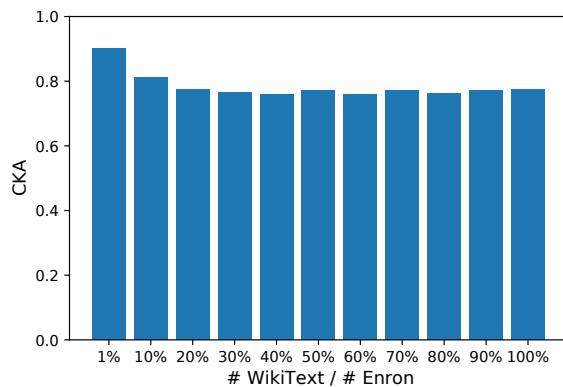
- [1] 2015. Enron Email Dataset. <https://www.cs.cmu.edu/~enron/>.
- [2] 2024. Ai4Privacy. <https://huggingface.co/datasets/ai4privacy/pii-masking-300k>.
- [3] 2024. Find Emails. <https://www.findemails.com/>.
- [4] 2024. findthatlead. <https://findthatlead.com/en/>.
- [5] 2024. Fine-tune a Llama 2 model in Azure AI Studio. <https://learn.microsoft.com/en-us/azure/ai-studio/how-to/fine-tune-model-llama>.
- [6] 2024. Fine-tuning - OpenAI. <https://platform.openai.com/docs/guides/fine-tuning>.
- [7] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.
- [8] Alan Akbik, Tanja Bergmann, Duncan Blythe, Kashif Rasul, Stefan Schweter, and Roland Vollgraf. 2019. FLAIR: An easy-to-use framework for state-of-the-art NLP. In *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. 54–59.
- [9] Alex Albert. 2023. Jailbreak Chat. <https://www.jailbreakchat.com/prompt/4f37a029-9dff-4862-b323-c96a5504de5d>
- [10] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022. Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback. *CoRR* abs/2204.05862 (2022). <https://doi.org/10.48550/arXiv.2204.05862> arXiv:2204.05862
- [11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.
- [12] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chi-yuan Zhang. 2022. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646* (2022).
- [13] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*. 2633–2650.
- [14] Ilias Chalkidis, Ion Androutsopoulos, and Nikolaos Aletras. 2019. Neural legal judgment prediction in English. *arXiv preprint arXiv:1906.02059* (2019).
- [15] Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. *arXiv preprint arXiv:2004.12651* (2020).
- [16] Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. *arXiv preprint arXiv:1902.10547* (2019).
- [17] Hyung Won Chung, Le Hou, Shayna Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling Instruction-Finetuned Language Models. *CoRR* abs/2210.11416 (2022). <https://doi.org/10.48550/arXiv.2210.11416> arXiv:2210.11416
- [18] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. 2023. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773* (2023).

- [19] Tyler L Hayes, Kushal Kafle, Robik Shrestha, Manoj Acharya, and Christopher Kanan. 2020. Remind your neural network to prevent catastrophic forgetting. In *European Conference on Computer Vision*. Springer, 466–483.
- [20] Shlomo Hoory, Amir Feder, Avichai Tendler, Sofia Erell, Alon Peled-Cohen, Itay Laish, Hootan Nakhost, Uri Stemmer, Ayelet Benjamini, Avinatan Hassidim, et al. 2021. Learning and evaluating a differentially private pre-trained language model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. 1178–1189.
- [21] Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. 2022. Are large pre-trained language models leaking your personal information? *arXiv preprint arXiv:2205.12628* (2022).
- [22] Ronald Kemker, Marc McClure, Angelina Abitino, Tyler Hayes, and Christopher Kanan. 2018. Measuring catastrophic forgetting in neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.
- [23] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*. PMLR, 3519–3529.
- [24] Haoran Li, Dadi Guo, Wei Fan, Mingshi Xu, and Yangqiu Song. 2023. Multi-step jailbreaking privacy attacks on chatgpt. *arXiv preprint arXiv:2304.05197* (2023).
- [25] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* 55, 9 (2023), 195:1–195:35. <https://doi.org/10.1145/3560815>
- [26] Xiaogeng Liu, Nan Xu, Muhan Chen, and Chaowei Xiao. 2023. Autodan: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451* (2023).
- [27] Nils Lukas, Ahmed Salem, Robert Sim, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. 2023. Analyzing Leakage of Personally Identifiable Information in Language Models. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 346–363.
- [28] Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. 2023. An empirical study of catastrophic forgetting in large language models during continual fine-tuning. *arXiv preprint arXiv:2308.08747* (2023).
- [29] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *International Conference on Learning Representations*.
- [30] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843* (2016).
- [31] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*. [http://papers.nips.cc/paper\\_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html](http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html)
- [32] Mustafa Ozdayi, Charith Peris, Jack FitzGerald, Christophe Dupuy, Jimit Majumdar, Haidar Khan, Rahil Parikh, and Rahul Gupta. 2023. Controlling the Extraction of Memorized Data from Large Language Models via Prompt-Tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Toronto, Canada, 1512–1521. <https://doi.org/10.18653/v1/2023.acl-short.129>
- [33] Kellin Pelrine, Mohammad Taufeeque, Michal Zajac, Euan McLean, and Adam Gleave. 2023. Exploiting novel gpt-4 apis. *arXiv preprint arXiv:2312.14302* (2023).
- [34] Renji Pi, Tianyang Han, Yueqi Xie, Rui Pan, Qing Lian, Hanze Dong, Jipeng Zhang, and Tong Zhang. 2024. MLLM-Protector: Ensuring MLLM’s Safety without Hurting Performance. *arXiv preprint arXiv:2401.02906* (2024).
- [35] Xiangu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. 2023. Fine-tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! *arXiv preprint arXiv:2310.03693* (2023).
- [36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.
- [37] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.
- [38] Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczęsła, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask Prompted Training Enables Zero-Shot Task Generalization. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25–29, 2022*. <https://openreview.net/forum?id=9Vrb9D0WI4>
- [39] Rylan Schaeffer, Brando Miranda, and Sammi Koyejo. 2023. Are emergent abilities of Large Language Models a mirage? *arXiv preprint arXiv:2304.15004* (2023).
- [40] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2023. “Do Anything Now”: Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. *arXiv preprint arXiv:2308.03825* (2023).
- [41] Xinyue Shen, Zeyuan Chen, Michael Backes, Yun Shen, and Yang Zhang. 2023. “Do Anything Now”: Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models. *CoRR abs/2308.03825* (2023). <https://doi.org/10.48550/arXiv.2308.03825>
- [42] Reza Shokri and Vitaly Shmatikov. 2015. Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 1310–1321.
- [43] Veri Sorin and Eyal Klang. 2023. Large language models and the emergence phenomena. *European Journal of Radiology Open* 10 (2023), 100494.
- [44] Zhiqing Sun, Sheng Shen, Shengcao Cao, Haofian Liu, Chunyun Li, Yikang Shen, Chuang Gan, Liang-Yan Gui, Yu-Xiong Wang, Yiming Yang, Kurt Keutzer, and Trevor Darrell. 2023. Aligning Large Multimodal Models with Factually Augmented RLHF. *CoRR abs/2309.14525* (2023). <https://doi.org/10.48550/arXiv.2309.14525>
- [45] Kusal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization Without Overfitting: Analyzing the Training Dynamics of Large Language Models. In *Advances in Neural Information Processing Systems*, Vol. 35. Curran Associates, Inc., 38274–38290. [https://proceedings.neurips.cc/paper\\_files/paper/2022/file/fa0509f4dab6807e2cb465715bf2d249-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2022/file/fa0509f4dab6807e2cb465715bf2d249-Paper-Conference.pdf)
- [46] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).
- [47] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Kashlykov, Soumya Batra, Prajwäl Bhargava, Shruti Bhowmik, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).
- [48] Pablo Villalobos, Jaime Sevilla, Lennart Heim, Tamay Besiroglu, Marius Hobbehahn, and Anson Ho. 2022. Will we run out of data? An analysis of the limits of scaling datasets in Machine Learning. *arXiv preprint arXiv:2211.04325* (2022).
- [49] Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Stevers, and William Yang Wang. 2023. Large language models are implicitly topic models: Explaining and finding good demonstrations for in-context learning. In *Workshop on Efficient Systems for Foundation Models@ ICML2023*.
- [50] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2024. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems* 36 (2024).
- [51] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned Language Models are Zero-Shot Learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25–29, 2022*. <https://openreview.net/forum?id=gEZrGCozdqR>
- [52] Yueqi Xie, Jingwei Yi, Jiawei Shao, Justin Curl, Lingjuan Lyu, Qifeng Chen, Xing Xie, and Fangzhou Wu. 2023. Defending chatgpt against jailbreak attack via self-reminders. *Nature Machine Intelligence* 5, 12 (2023), 1486–1496.
- [53] Qiusi Zhan, Richard Fang, Rohan Bindu, Akul Gupta, Tatsunori Hashimoto, and Daniel Kang. 2023. Removing rlhf protections in gpt-4 via fine-tuning. *arXiv preprint arXiv:2311.05553* (2023).
- [54] Zhixin Zhang, Jiaxin Wen, and Minlie Huang. 2023. ETHICIST: Targeted Training Data Extraction Through Loss Smoothed Soft Prompting and Calibrated Confidence Estimation. *arXiv preprint arXiv:2307.04401* (2023).
- [55] Rui Zhu, Di Tang, Siyuan Tang, XiaoFeng Wang, and Haixu Tang. 2022. Selective Amnesia: On Efficient, High-Fidelity and Blind Suppression of Backdoor Effects in Trojaned Machine Learning Models. *arXiv preprint arXiv:2212.04687* (2022).

## Appendix A

### A.1 Analysis on Catastrophic Forgetting

We further investigated the phenomenon of catastrophic forgetting during the simulation process in Section 4.2. In Figure 6, we examined the training process to obtain  $f_{forget}$ , with varying proportions of WikiText data employed in the training dataset, by evaluating the CKA (Centered Kernel Alignment) similarity between the penultimate layer of the model pre and post-training on the PII association task data  $\mathcal{S}$ . Our results revealed that catastrophic forgetting happens quickly with the ratio of WikiText to Enron data increases from 1% to 10%, i.e., the CKA similarity significantly decreases



**Figure 6: CKA on the penultimate layer of GPT2-small trained with different ratios of non-privacy data.**

when the model continues to learn other tasks. However, when the ratio of WikiText to Enron data continue to increase beyond 20%, the CKA similarity essentially stabilizes, indicating a cessation in further alterations. This suggests that even during the LLM training phase, with the incorporation of an increased volume of non-PII association task data, the feature space pertaining to the PII association task undergoes minimal alterations. This substantiation underpins the validity of our experimental simulation.