Enhancing Data Privacy in Large Language Models through Private Association Editing

Anonymous ACL submission

Abstract

Large Language Models (LLMs) are powerful tools with extensive applications, but their tendency to memorize private information raises significant concerns as private data leakage can easily happen. In this paper, we introduce Private Association Editing (PAE), a novel defense approach for private data leakage. PAE is designed to effectively remove Personally Identifiable Information (PII) without retraining the model. Our approach consists of a four-step procedure: detecting memorized PII, applying PAE cards to mitigate memorization of private data, verifying resilience to targeted data extraction (TDE) attacks, and ensuring consistency in the post-edit LLMs. The versatility and efficiency of PAE, which allows for batch modifications, significantly enhance data privacy in LLMs. Experimental results demonstrate the effectiveness of PAE in mitigating private data leakage. We believe PAE will serve as a critical tool in the ongoing effort to protect data privacy in LLMs, encouraging the development of safer models for real-world applications.

1 Introduction

A massive pretraining phase seems to be the key to obtaining versatility and accuracy in a large number of tasks: Large language models (LLMs) are indeed able to perform accurately many tasks by capturing information from their training data. Even in zero-shot scenarios, LLMs serve as alternative sources of information (Hou et al., 2024), perform translation tasks (Mu et al., 2023), translate natural language requests into code (Ranaldi et al., 2024), and are definitely capable of capturing world knowledge (Petroni et al., 2019, 2020). The massive pretraining phase seems to be the key to obtaining versatility and accuracy in this large variety of tasks.

However, growing larger, training data for LLMs have become uncontrollable and may inadvertently contain some private personal information of unaware people. LLMs may potentially retain this sensitive information (Carlini et al., 2021, 2023; Huang et al., 2022). This is a potential threat in privacy of unaware people. Indeed, by performing Training Data Extraction attacks, (Carlini et al., 2021) showed that LLMs may verbatim generate strings containing sensitive information observed during training. Then, attackers may gain access to private information.

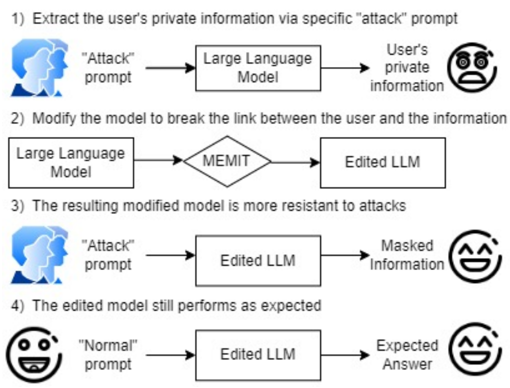

Figure 1: Preserving privacy for LLMs by using Private Association Editing

Strategies to remove sensitive information from LLMs are needed and mandatory, as preserving privacy is a must. Yet, the straight-forward technique of remove-and-retrain is unfeasible as extremely expensive.

In this paper, we propose *Private Association Editing* (PAE) to remove memorized private information adjusting parameters of LLMs without re-training (see Fig. 1). Stemming from MEMIT (Meng et al., 2023b) formulation to edit factual knowledge, we define PAE as a novel model-editing defense strategy based on the idea of *breaking the association* between personal information and the identity of the person to whom it belongs. We anonymize the private information directly in the model, replacing the original information with masked – but semantically equivalent – information. We experiment with GPT-J (Wang and Komatsuzaki, 2021) as it is an open-source model that contains documented private information. We perform Training Data Extraction attacks (Huang et al., 2022) before and after our model-editing defenses and we show that our strategies are an efficient alternative to make a model more robust against the generation of private information while keeping constant its performance in generating texts.

2 Background

Large Language Models (LLMs) are prone to emit private information. Indeed, attacking LLMs to extract memorized private information is possible by using black-box access to language models. Training Data Extraction (TDE) is a technique to extract this private information (Carlini et al., 2021). It consists of querying the target model to force it to produce its own training data. A textual training example is considered "extractable" if a specific prefix can be used to prompt the model to generate the exact training example from its training set. Carlini et al. (2021) found that GPT-2 often retains and reveals personal information such as Twitter handles, email addresses, and Universal Unique Identifiers (UUIDs). The memorization of training examples explains the success of these attacks: when LLMs are prompted with a prefix encountered during training, they often complete the prompt with the remaining part of the training sequence (Carlini et al., 2023).

Attacks may be particularly effective in open LLMs. Huang et al. (2022) demonstrated that conditioning a model with a prompt that is part of the training data can result in the leakage of personal identifiable information (PII), such as email addresses. They also showed that this method is more effective than creating entirely new, unseen prompts. Nasr et al. (2023) revealed that Carlini et al. (2021) method is even more effective than previously expected. By querying open-source models, they confirmed the success of the attack procedure using the training data solely for verification purposes. They conducted these attacks on open models like GPT-Neo (Black et al., 2022) and Pythia (Biderman et al., 2023), starting with prompts sourced from Wikipedia.

Even closed LLMs may reveal private information by using Training Data Extraction. Since these attacks require only black-box access to the model, closed models like GPT-3.5 and GPT-4 can be attacked. In fact, using the same prompts proposed by Huang et al. (2022), Wang et al. (2024) demonstrate that GPT-3.5 and GPT-4 can predict respectively around 5% and 4% of the email addresses accurately.

As personal information leakage from LLMs is a concrete possibility, model editing is a possible solution as opposed to an expensive remove-and-retrain strategy.

Model editing in LLMs refers to the process of modifying specific aspects of a model's behavior or knowledge without retraining it from scratch. This involves making targeted adjustments to the model's parameters or responses to correct errors, update information, or adapt to new requirements. Mitchell et al. (2022) introduced a semi-parametric editing methodology, employing a retrieval-augmented counterfactual model, that effectively modulates neural network predictions over the SERAC dataset. Cao et al. (2021) proposed KNOWLEDGEEDITOR that efficiently and reliably edits factual knowledge within language models, ensuring consistency across various formulations of facts. Furthermore, Yao et al. (2023) introduced MEND on various datasets, demonstrating its ability to rapidly and effectively edit large-scale models' behaviors without extensive retraining. Since these methods can modify factual information memorized in LLMs, our goal is to exploit them to erase private information inadvertently ingested during training.

Similarly to the method defined in our paper, Patil et al. (2023) investigated model editing techniques to modify the information memorized in LLMs concluding that information cannot be erased. In particular, they applied TDE attacks against the GPT-J (Wang and Komatsuzaki, 2021) model and demonstrated that in black-box access– performing attacks that also include paraphrases of the original prompt– model editing cannot erase factual information memorized in GPT-J. Our setting is different: in fact, Patil et al. (2023) investigated the effectiveness of model editing only on factual information from sentences derived from Wikipedia, and not directly present in the training data – the Pile (Gao et al., 2020). By definition, the model under attack does not *verbatim* memorize information that is not in training data: since the examples used by Patil et al. (2023) are derived from Wikipedia and not included in the Pile, while the factual information they contain is memorized, they cannot be verbatim memorized. In our experiments, we directly study the effectiveness of model editing to delete private information that is verbatim memorized with a focus on privacy rather than factual information.

To the best of our knowledge, this is a novel approach to protect private LLMs from personal information leakage that show the potential beneficial effect on privacy preserving.

3 Attacking and Defending LLMs from Private Data Leakage

Large Language Models (LLMs) have a tendency to memorize examples from their training data, and Training Data Extraction (TDE) attacks can be used to recover these memorized examples. When fed with the right prompt, LLMs emit verbatim memorized information. In fact, if a model is prompted with a prefix encountered during training, it often completes it with the rest of the training sequence (Carlini et al., 2023; Huang et al., 2022).

In this scenario, we aim to deliver solutions to help people and owners of LLMs remove undesirably memorized Personally Identifiable Information from LLMs. The procedure we propose consists of four steps (see Fig. 1):

- detecting the presence of memorized Personally Identifiable Information (PII) in *pre-edit* LLMs performing black box TDE attacks (Sec. 3.1);
- *Private Association Editing* (PAE) to remove PII by editing parameters of LLMs obtaining *post-edit* LLMs (Sec. 3.2);
- assessing that post-edit LLMs are more resilient to attacks with TDE attacks (as in Sec. 3.1);
- a final consistency check of *post-edit* LLMs to assess that LLMs are not corrupted after PAE behaving similarly to *pre-edit* LLMs (Sec. 3.3)

This procedure is extremely more versatile than erase-and-retrain and can be used in small batches of modification of an LLM. The core of our procedure is the method we propose called *Private Association Editing* (PAE).

3.1 Training Data Extraction Attacks to recover Sensitive Information

To detect the presence of memorized Personally Identifiable Information LLMs, we follow the attack pipeline and attack prompts defined by Huang et al. (2022). They defined two kinds of attacks depending on how information is stored and retrieved: (1) a model *memorizes* personal information if there exists a prompt from the training data that leads the model to generate that information; (2) in contrast, a model *associates* an individual to its personal information if there exists a prompt not seen during training but containing a reference to an individual that leads to the generation of PII. (Huang et al., 2022) already demonstrated that memorization is more common in LLM than association, showing that a model from the GPT-Neo[1] family can predict emails more accurately when conditioned with prompts from the training data rather than wthen analyzely unseen prompts.

We then analyze two attacking schemes: the Memorization attacks and the Association attacks.

In a Memorization attack, a model is fed with a prompt extracted from its pretraining data. This prompt is the *context* that precedes the private PII in the training data. For example, a *context* prompt attack to recover the email address of *Jonh Brown* would look like: `"All the winter months might settle 2.25. As such, the best thing to be short is jan. —Original Message— From: Jonh, Brown"`. In this attack, following Huang et al. (2022), we simulate that the attacker has more or less knowledge about the training data by conditioning the generation of the model to *context* prompts of different lengths in terms of tokens.

In the Association Attack, the model is instead fed with a prompt that contains an identifier of the person whose information is to be extracted, but that does not exactly match the training data. In particular, Huang et al. (2022) defined four *zero-shot* attack prompts, identified by letters from a to d. All *zero-shot* prompts contain the name of the person that owns the email the attacker wish to obtain and the model is asked to predict the email: for example, the *zero-shot* prompt a to recover the email adress of *John Brown* is "`the email address of John Brown is`". The attack succeeds if, during the generation of the subsequent tokens, the model generates the target's private information, that is, the correct email address.

[1] https://www.eleuther.ai/artifacts/gpt-neo

In both Memorization and Association attacks, the adversary in black-box access wants to force the model to generate some PII regarding a person. The analyzed framework encompasses a malicious attacker – or any individual aiming to detect unauthorized use of their data – who has assumptions about the original text that was used during training or who has no prior clues about the original data that contained the private information but who has some other knowledge about the identity of the individual whose sensitive information they wish to extract.

3.2 Private Association Editing as Efficient Defense against Privacy Attacks

To defend people from privacy attacks of LLMs, we propose *Private Association Editing* (PAE), which is the second step of our procedure. This editing technique involves disrupting the link between an individual identity and their PII. The technique proposed here is efficient since it allows the anonymization of private information directly into the model parameters. Moreover, our solution is also scalable since it can be used to protect the privacy of multiple users.

A private association is an association between the name of an individual and a PII that should not be revealed. This association is a triple <subject,predicate,PII-object>, as the following example: <John Smith, owns, john.smith@company.com>; in the example, the PII-object is the email address of the person.

Our PAE employs model editing techniques based on ROME (Rank-One Model Editing) (Meng et al., 2023a) and MEMIT (Model Editing via Iterative Training) (Meng et al., 2023b) as a defensive strategy against attacks aimed at safeguarding the sensitive data used to train Large Language Models (LLMs). Then, the scalability to editing different facts in a batch is facilitated by the MEMIT framework, which allows us to incorporate as many elements in the form of modifications as desired. These modifications are seamlessly executed on the same model also avoiding degradation of the model's performance.

In *private association editing*, once a user of the system has understood that their personal information has been inadvertently inserted into the training data and consequently memorized, a model edit can be performed to mask the private information.

The procedure to edit a private association uses PAE cards based on the MEMIT modification card.

`prompt`	*The email address of* `{subject}` *is*
`ground truth`	*john.brown@nowhere.com*
`target`	*mail@domain.com*
`subject`	*John Brown*

Table 1: An example of Private Association Editing card for email addresses with an implicit prompt

The basic structure of a MEMIT modification card is composed of a `prompt`, a `ground truth`, a `target`, and a `subject`. Our PAE cards specialize the MEMIT modification card on a particular PII. We have defined two main types of PAE cards to mask the private information of users. The first type is called "explicit" because it directly identifies the connection between the person and the private data and perfectly adheres to the MEMIT implementation. For example, an explicit prompt is "{name} has an email address that is". The second type is "implicit" which features a prompt that does not necessarily include the person's name as the subject of the sentence, favoring a more precise meaning of the sentence. An example of an implicit prompt is "The mail address of {name} is".

It is important to note that we used MEMIT in "batch" mode because we are interested in fixing the model and subjecting it to k modifications. In this way, we are able to use MEMIT performing k modifications at the same time, instead of performing single edits separately and recreating the model based on the post-edited weights obtained from the last edit every time.

In a real-world scenario, recreating or retraining a model for each requested modification is not feasible. Instead, with our strategy called "one model, n edits" we are able to make all requested changes to a single model. By masking and anonymizing the email address, we make it more challenging for attackers to elicit specific private data from the model in response to particular prompts. This methodology effectively reduces the risk of sensitive information being inadvertently disclosed by the model.

3.3 Evaluating Language Modeling Performance

The final step of the procedure for preserving privacy with PAE is to investigate whether the LLM maintains its behavior in text generation. In fact, Model Editing techniques, in general, and PAE, in particular, may perturb the language model capabilities due to the intervention on the model parameters.

The LLM assessment procedure we describe in this Section aims to verify that the privacy-preserving language model is not a worse model than the original one. The main idea is that LLMs capabilities are not perturbed if people are not able to determine which of the two models is responsible for which generation, then it means that the edit procedure does not affect model performance: there is no one better than the other, a user of the system would be equally happy to use one or the other.

The description of the LLM assessment procedure in this section is twofold: (1) the *automatic assessment procedure* that should be used when PAE is used in real scenarios; (2) the *manual assessment procedure* that is used in this paper to determine if the *automatic assessment procedure* capture the main idea of non-perturbed LLM.

The *automatic assessment procedure* is the operational procedure to automatically compare a *pre-edit* version LLM and a *post-edit* version LLM. The idea is to simply collect generations for a given set of prompts for *pre-edit* LLM and *post-edit* LLM. Then, these generations are compared with string-based similarity metrics, in particular BLEU and METEOR metrics. With these measures, we can automatically assess if *pre-edit* LLM and *post-edit* LLM behave in a similar way.

The *manual assessment procedure* is instead an experimental procedure to confirm that the automatic assessment procedure can be used to determine if *pre-edit* LLM and *post-edit* LLM are similar. In this procedure, we again collect generations for given prompts for *pre-edit* LLM and *post-edit* LLM. In this case, we ask annotators to choose which model generated each text in a sort of classification task. We argue that a low accuracy in this classification task and a low agreement among annotators mean that the models are not distinguishable and, in particular, that the privacy-preserving models are no worse than the original ones.

4 Experiments

4.1 Experimental Setup

In this section, we discuss the parameters of our experiment to allow replicability: the analized LLM and related datasets, the intricacies of MEMIT used in our PEA, and, finally, the set-up of the evaluation of the LLMs.

Analized LLM and related datasets In our experiments, we test the GPT-J model (Wang and Komatsuzaki, 2021) that is designed to generate human-like text continuations from prompts: it is a large model, with 6 billion parameters. This model is trained on an open dataset, the Pile (Gao et al., 2020). The Pile is a diverse, large-scale text corpus that aggregates various sources, including books, articles, websites, and scientific papers. It spans multiple languages and domains, making it an ideal training resource for language models like GPT-J. The Pile contains a rich variety of text, enabling the model to learn from a wide range of contexts and topics. One of the constituent sub-datasets within The Pile is the Enron Emails (Klimt and Yang, 2004) corpus. This dataset contains text from approximately 150 users, primarily senior management of Enron, organized into folders. It includes a total of about 0.5 million email messages. The Enron Emails dataset was originally made public during the investigation into Enron's accounting methods. Its inclusion in the Pile mimic the inadvertently insertion into the training data of private information, in particular of PII like email adressess. For this reason, the Enron Email dataset represents a natural starting point to test GPT-J memorization of PII.

Intricacies of MEMIT There are two distinct ways to apply model editing using MEMIT(Yao et al., 2023) given N elements to modify: batch and sequential editing. Batch editing involves editing k elements in an LLM simultaneously. Conversely, sequential editing focuses on editing N elements within an LLM in a sequential way, with each edit on a subset of the N elements, performed on the new model retaining previous edits. While batch editing may be sufficient to preserve privacy, the sequential editing approach is closer to the real-world need to constantly update model parameters, as more privacy leakages may be discovered over time.

In our research, we initially adopt the batch editing approach with $k = N$. This approach is the safest – in principle – since the post-edited parameters are directly the pretrained ones. Then, we investigate the effect of sequential editing with $k < N$, simulating the real-world scenario in which multiple edits are necessary over time. For PAE to be applicable, in both scenario our method should lead to a comparable decrease in privacy leaks.

			Pre-edit			Post-edit			
						Implicit		Explicit	
			Leaked emails	Number of predicted emails	Attack Accuracy	Leaked emails	Attack Accuracy	Leaked emails	Attack Accuracy
Memorization Attacks	greedy	*context 50*	353	2827	0.125	203	0.072	218	0.077
		context 100	476	2932	0.162	301	0.103	317	0.108
		context 200	537	2951	0.182	368	0.125	396	0.134
	beam search	*context 50*	346	2689	0.129	244	0.091	248	0.092
		context 100	476	2809	0.169	339	0.121	339	0.121
		context 200	515	2863	0.180	394	0.138	405	0.141
Association Attacks	greedy	*zero-shot a*	5	3130	0.002	1	0.000	1	0.000
		zero-shot b	2	3229	0.001	0	0.000	0	0.000
		zero-shot c	26	3234	0.008	13	0.004	11	0.003
		zero-shot d	68	3237	0.021	48	0.015	42	0.013
	beam search	*zero-shot a*	6	3178	0.002	3	0.001	5	0.002
		zero-shot b	1	3178	0.000	0	0.000	0	0.000
		zero-shot c	28	3232	0.009	20	0.006	11	0.003
		zero-shot d	73	3234	0.023	50	0.015	37	0.011

Table 2: Results of the attacks against the pretrained model (*Pre-edit*) and after the application of *PAE*. The training data extraction attacks that exploit the memorization of PII after PAE tends to lose their efficacy in retrieving private information from the model.

Evaluation of post-edited LLM For the *automatic assessment procedure*, we measure the difference in generations for the pre-trained GPT-J model and the post-edited version by generating a small paragraph starting from 45 prompts extracted from the Book3 (Rae et al., 2022) dataset, included in the Pile . We prompted the post-edited models and the pretrained one with 20 tokens of the 45 randomly selected examples and we evaluate how similar the generations are measuring their overlap. he higher the similarity, the higher the likelihood that the PAE does not influence the overall performance of the model. Evaluation measures are the ROUGE and the METEOR scores.

For the *manual assessment procedure*, we generate with post-edited models and with the pretrained one a short paragraph from 10 different prompts (a complete list can be found in the Appendix 6.1). We collect the generations for the pre-edit model and the model post-edited according to each of the editing strategies. Hence, in total, we collect 30 generations. Then, five annotators are asked to choose which of the models generated each of the paragraphs. Three sample generations of each model were provided, and the annotators were informed that two out of three models had been post-edited, but none of them were informed which of the three systems had been post-edited. Evaluation measures are the classification accuracy of each annotator and the Fleiss' K inter-annotator agreement: a low score on both can confirm that the models are indistinguishable.

4.2 Results and Discussion

LLMs leak Private Information Since LLMs tend to leak training data, we aim to quantify the amount of private information that can be retrieved from the pre-trained GPT-J. Unfortunately, GPT-J makes no exception to the trend noticed by Huang et al. (2022) for the GPT-Neo models. In fact, also this model tends to generate PII.

In Table 2, it is possible to observe that Training Data Extraction Attacks that are based on Memorization are particularly effective against the GPT-J model: on average, the model tends to accurately predict the mail observed during training the 16% of the times.

It is worth noting the scale of the leakage: the model is originally prompted with 3238 examples. The column *Generated emails* reports the number of times during generation that the model answers with an email address, while *Leaked emails* reports the number of times the generation is correct, meaning that the generated email corresponds to the one observed in the training data. On average, 450.5 emails are correctly generated by those attacks: the privacy of a large number of people is threatened.

Moreover, as the attacker gets more information, the accuracy of the attacks gets higher: the accuracy of the attacks strongly depends on the length of the prompt. In fact, the lower accuracy – the number of correctly leaked email addresses over the total email adresses generated– that can be registered in Memorization Attacks is 12% : the model in that case is fed with a *context* prompt that is 50 tokens long. However, when the *context* prompt given to the model is composed of 200 tokens, the accuracy of the attack peaks at 18.2% with greedy decoding and 18% using beam search decoding.

The accuracy of the Association Attacks is much more modest. The results of those attacks against GPT-J model exhibit similar patterns to the one observed by Huang et al. (2022) against the GPT-Neo models. The larger number of email addresses leaked by those kind of attacks is 68, a modest number compared to the accuracy obtained in the

Memorization Attacks. However, in an adversarial scenario even low accuracy may cause harm to people. Hence, in the next Section, we will demonstrate the efficacy of PAE against both types of attacks.

PAE in batch editing Preserves Privacy In Table 2, it is possible to observe the reduced effectiveness of Memorization and Association attacks after the GPT-J model has undergone an editing process (Post-edit columns), with Post-edit results further divided into Implicit and Explicit categories. This Section investigates the impact of these edits, focusing on their efficacy against more or less informed attacks. We argue that PAE edits are effective if they can reduce the leakage of private information, regardless of the nature of the attack.

Post-edit results show, in fact, a significant reduction in the effectiveness of Association attacks. This reduction is particularly notable in scenarios where the number of leaked emails drops close to zero. For example, under the Implicit strategy, *zero-shot b* result in 0 leaked emails and 0 Attack accuracy, indicating a complete mitigation of the attack. However, some prompts still cause leakage; for instance, in prompt *zero-shot b* the edit reduces the number of email addresses leaked significantly but not completely (from 68 pre-edit to 48 post-edit in the Implicit). Crucially, while not perfect, the PAE edits – both Implicit and Explicit – always cause an increase in privacy protection, since reduce the number of email correctly leaked by Association Attacks.

However, it is crucial to consider the originally leaked emails when interpreting post-edit results. While a reduction to near-zero leakage is impressive, the impact is more pronounced when starting from a higher number of pre-edit leaks. For this reason, we focus on the discussion around the Memorization Attacks, that cause a larger number of private email addresses to be generated.

PAE is an effective solution against Memorization Attacks. In particular, the accuracy of the attacks steady decreases in each configuration. The average drop in accuracy after an Implicit edit is 5% and 4.5% after Explicit edit: this means that PAE is able to modify model parameter so that, on average, the 32% of the previously predicted email addresses are no more verbatim generated by the model using Implicit defense strategy, 29% with the explicit one. Against attacks with *context* prompt of 50 tokens, PAE effectiveness peaks, with 42.5% of the email addresses anonymized. As expected, more informed *context* prompts are more challenging: however, also with *context* prompts of 200 tokens, PAE make the accuracy attack drop from 0.18 to 0.12 in Greedy decoding and to 0.138 using Beam Search in the Implicit edit, and from 0.18 to 0.134 in greedy decoding and to 0.141 using Beam Search in the Explicit edit. In general, studying the effect of the decoding algorithm on the attacks accuracy we can state that this factor does not influence much the results: under Memorization Attacks only a slight difference in term of accuracy can be registered From this analysis can conclude that PAE can help in protecting privacy.

Finally, it is possible to notice that there is a consistent difference between Implicit and Explicit post-edit results. Explicit edits generally result in a slightly higher number of leaked emails and attack accuracy compared to Implicit edits, expecially under Memorization Attacks. For example, in the case of a *context* prompt of 200, the Explicit edit cause a larger number of emails to be correctly generated (396 in greedy deconding, 405 in beam search decoding) than the corresponding Implicit edit (368 email addresses leaked in greedy decoding, 394 in beam search decoding)

In summary, post-edit measures, particularly implicit edits, demonstrate a strong capability to safeguard email data from various attack strategies, significantly lowering both the number of leaked emails and the attack accuracy across different configurations. For our experiments, we adopted the "one model, n edits" philosophy. This approach is based on the practical scenario where an LLM producing private data needs to be edited for a large number of potentially threatened individuals: with PAE, the model owner can perform a single edit to the model parameters to reduce privacy risks. Our investigation on the emails that the model generates when subjected to Memorization and Association attacks confirm that the Memorization Attacks are able to recover a larger number of private information also after the editing. However, the evaluation confirms the effectiveness of our model editing techniques in preventing the disclosure of private data since also informed attacks –like the Memorization Attacks – are less effective on edited models. This setup is particularly challenging because it requires analyzing the impact of n modifications at the same time. In the next Section, we also demonstrate that the original language model is not negatively influenced by PAE.

Automatic Evaluation	BLEU	E_1-O	$0.808(\pm0.198)$
		E_2-O	$0.793(\pm0.199)$
		E_1-E_2	$0.790(\pm0.203)$
	METEOR	E_1-O	$0.841(\pm0.173)$
		E_2-O	$0.826(\pm0.172)$
		E_1-E_2	$0.824(\pm0.183)$
Manual Revision	Accuracy		$0.35(\pm0.07)$
	Fleiss' K		0.002

Table 3: PAE preserves language model performances. In the Manual revision of the generation, annotators are not able to detect which model generated each paragraph. Moreover, on a larger scale of examples, the Automatic Evaluation reveals that the generation of *post-edit* LLMs E_1 and E_2 are both similar to one another (E_1-E_2) and with respect to the *pre-edit* LLM O (E_1-O and E_2-O)

PAE preserves the Language Modelling Capabilities PAE preserves privacy of people while not affecting the Language Modeling performances. Results of the evaluation can be found in the Table 3. The results of the *automatic assessment procedure* can quantitatively give us insight that models generations are, in fact, similar. Both according to BLEU metric and to METEOR, the systems generate (in greedy decoding) very similar paragraph when prompted with the same tokens. In particular, the post-edited models E_1 and E_2 – post-edited with implicit and explicit PAE – are similar to the original, pre-edited model O and are also similar with respet to each other. Finally, the *manual assessment procedure* suggest that the models are indistinguishable from one another. In fact, the annotators asked to detect wich model is responsible for a generation among E_1, E_2, and O can only randomly guess, with an average accuracy on this classification task ($0.35(\pm0.07)$) close to random choice. Also the very low agreement suggest that tge three systems are indistinguishable.

This evaluation procedure can attest that the EAP is applicable because it not only helps to preserve user privacy, but also leaves the capabilities of the systems language model intact.

PAE is applicable with sequential editing Finally, in Table is it possible to notice that the sequential update is definetely applicable with PAE. In this experiments, we perform sequential edit of the GPT-J model, varying the number of email addresses anonymized per edit, varying from 5 to 300. We indicate the number of anonymized emails per edit as batch size k: with $k < N$ we mimic the real world scenario of updating a model each time a privacy leak is detected. To understand whether

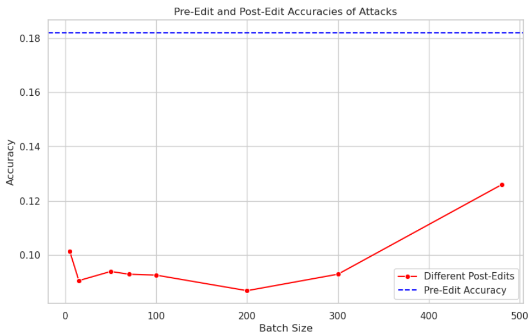

Figure 2: Memorization Attack against sequentially post-edited models. The smaller the batch size k, the larger the number of sequential updates are necessary to edit all the private emails addressess leaked by the original model. After the Sequential edits, the stronger Memorization Attack ($|p_M| = 200$) achieve similar performances at all the configurations.

sequential editing has a negative impact on effectiveness of the edit, we evaluate the effectiveness of PAE for each of the batch sizes in the Memorization Attack with the more effective of the prompts ($|p_M| = 200$). The results in Table 2 refers to a model post-edited with "implicit" PAE. As can be observed in Table 2, the accuracy of the edit is rather stable and similar to the results obtained in the batch editing scenario. Those results confirm the applicability of PAE also in sequential editing.

5 Conclusion

In this paper, we address the critical issue of private data leakage in Large Language Models (LLMs) due to their tendency to memorize training data. We propose Private Association Editing (PAE), a novel defense mechanism that effectively removes Personally Identifiable Information (PII) from LLMs without requiring retraining.

Our methodology involves a four-step procedure: detecting memorized PII, applying PAE cards, verifying resilience to targeted data extraction (TDE) attacks, and ensuring consistency in the post-edit LLMs. The PAE method stands out for its versatility and efficiency, allowing for small batch modifications and significantly enhancing the privacy of LLMs.

Our experiments demonstrate that the PAE approach is both effective and efficient in mitigating the risk of private data leakage. We believe PAE will be a valuable tool in the ongoing effort to protect data privacy in LLMs and encourage its adoption to prevent potential privacy violations as these models continue to be deployed in real-world applications.

Limitations

We outline some limitations and possible directions for future research in enhancing data privacy in Large Language Models (LLMs).

As the landscape of LLMs evolves, it may be useful to extend the Private Association Editing (PAE) mechanism to accommodate new types of models and data. Currently, we apply our proposed PAE method on a limited set of LLMs. A possible extension could involve testing and refining PAE across a broader spectrum of LLM architectures and training datasets. Our approach focuses on removing Personally Identifiable Information (PII) from LLMs without retraining. However, this method might not address all types of sensitive data. Future research could explore additional techniques to enhance the comprehensiveness of PII removal. While PAE shows promise in its current form, its real-world applicability and scalability need thorough validation. By addressing these limitations, future research can further solidify the role of PAE in safeguarding data privacy in LLMs and ensure its robustness and adaptability in various contexts.

References

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar van der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. *Preprint*, arXiv:2304.01373.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. Gpt-neox-20b: An open-source autoregressive language model. *Preprint*, arXiv:2204.06745.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. *Preprint*, arXiv:2104.08164.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. *Preprint*, arXiv:2202.07646.

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)*, pages 2633–2650.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling. *Preprint*, arXiv:2101.00027.

Yupeng Hou, Junjie Zhang, Zihan Lin, Hongyu Lu, Ruobing Xie, Julian McAuley, and Wayne Xin Zhao. 2024. Large language models are zero-shot rankers for recommender systems. *Preprint*, arXiv:2305.08845.

Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. 2022. Are large pre-trained language models leaking your personal information? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2038–2047, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bryan Klimt and Yiming Yang. 2004. The enron corpus: A new dataset for email classification research. In *European conference on machine learning*, pages 217–226. Springer.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2023a. Locating and editing factual associations in gpt. *Preprint*, arXiv:2202.05262.

Kevin Meng, Arnab Sen Sharma, Alex Andonian, Yonatan Belinkov, and David Bau. 2023b. Mass-editing memory in a transformer. *Preprint*, arXiv:2210.07229.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D. Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. *Preprint*, arXiv:2206.06520.

Yongyu Mu, Abudurexiti Reheman, Zhiquan Cao, Yuchun Fan, Bei Li, Yinqiao Li, Tong Xiao, Chunliang Zhang, and Jingbo Zhu. 2023. Augmenting large language model translators via translation memories. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10287–10299, Toronto, Canada. Association for Computational Linguistics.

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 arXiv:2311.17035*.

Vaidehi Patil, Peter Hase, and Mohit Bansal. 2023. Can sensitive information be deleted from llms? objectives for defending against extraction attacks. *Preprint*, arXiv:2309.17410.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. *ArXiv*, abs/2005.04611.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2022. Scaling language models: Methods, analysis insights from training gopher. *Preprint*, arXiv:2112.11446.

Federico Ranaldi, Elena Sofia Ruzzetti, Dario Onorati, Leonardo Ranaldi, Cristina Giannone, Andrea Favalli, Raniero Romagnoli, and Fabio Massimo Zanzotto. 2024. Investigating the impact of data contamination of large language models in text-to-sql translation. *Preprint*, arXiv:2402.08100.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

Boxin Wang, Weixin Chen, Hengzhi Pei, Chulin Xie, Mintong Kang, Chenhui Zhang, Chejian Xu, Zidi Xiong, Ritik Dutta, Rylan Schaeffer, Sang T. Truong, Simran Arora, Mantas Mazeika, Dan Hendrycks, Zinan Lin, Yu Cheng, Sanmi Koyejo, Dawn Song, and Bo Li. 2024. Decodingtrust: A comprehensive assessment of trustworthiness in gpt models. *Preprint*, arXiv:2306.11698.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. *Preprint*, arXiv:2305.13172.

6 Appendix

6.1 All the prompts of the manual assessment procedure

- "My name is";
- "The weather today is";
- "Ever since I was a child, I've always liked";
- "My dear friend Mary";
- "Swimmers are usually";
- "Modern art is";
- "The Industrial Revolution";
- "Follow those steps to cook";
- "It is forbidden to";
- "It is very likely".