Protecting the Privacy in Velvet with Model Editing

Giancarlo A. Xompero[1,2,*], Elena Sofia Ruzzetti[2], Cristina Giannone[1], Andrea Favalli[1], Raniero Romagnoli[1] and Fabio Massimo Zanzotto[2]

[1]Almawave S.p.A., Via di Casal Boccone, 188-190 00137, Rome, Italy
[2]Human Centric ART, University of Rome Tor Vergata, Italy

Abstract

Large Language Models (LLMs) showed impressive generation abilities and are now integrated in many real-world applications. However, LLMs also tend to memorize information, including Personally Identifiable Information (PII), which can be learned and generated during inference, posing a risk to users' privacy. In this context, Model Editing techniques have been proposed recently to prevent the leakage of private information by modifying LLMs' parameters directly while preserving their generation capabilities. In this work, we show an application of Model Editing for Privacy Protection in the context of Italian data on Velvet, a multilingual LLM recently released. In particular, we focus on protection against Training Data Extraction (TDE) attacks. Empirical results from the experiments show that model editing techniques can be effective in mitigating privacy leakage in LLMs, even for Italian data, while preserving their multilingual generation capabilities.

Keywords

Large Language Models, Model Editing, Privacy

1. Introduction

Large Language Models (LLMs) showed impressive generation capabilities in managing various tasks, and they are now integrated into many real-world applications. Given the popularity and potential of these models, several open-weight LLMs have been released to the public in the last years, including multilingual ones. Following this trend, LLMs that support the Italian language have also been made available [1], thus allowing to manage tasks even in Italian.

However, since LLMs are now employed in many services, they can be affected by some well-known issues, such as toxicity or privacy leakage [2], which can have an important negative impact on model performance. These problems raised concerns about privacy due to the possible presence of undetected private information in training data. Prior research showed that these models tend to memorize training data [3, 4, 5, 6], thus they are prone to memorizing Personal Identifiable Information (PII), which might be disclosed during the text generation. Italian LLMs can also be affected, as data used for training these models is often scraped from public web pages [7, 8], and although processes to identify and remove private information are used to clean data, PII could still be present.

Privacy is critical for LLMs deployed as services, raising concerns about privacy leakage and thus requiring attention. Carlini et al. [3] showed that extracting private information from an LLM is possible by prompting textual sequences from training data. The success of these attacks is evidence that the privacy of real individuals is at risk, so methods to prevent leakage of PII are necessary. Recently, many solutions have been proposed to mitigate this phenomenon, such as machine unlearning [9, 10]. Alternatively, Model Editing approaches showed promising effects for protecting the privacy of users [11, 12, 13]. The application of these methods allows us to modify the knowledge encoded in the LLMs by breaking the association between some memorized prompts and the corresponding PII. Among these methods, Private Memorization Editing (PME) [13] is an approach that exploits the memorization mechanism of transformers to modify the association between a prompt and its related private information, showing its effectiveness in protecting LLMs from TDE attacks.

In this work, we show an application of PME [13] to protect users' privacy for Italian data in LLMs. We focus on Velvet-2B[1], a recent multilingual LLM for English and Italian languages. Even though the training data has been curated to remove PII, the model may learn some information during training. Our main objective is to understand whether model editing can be extended and used to protect users' privacy whose PII might be included in training data obtained from public datasets. With PME, we can define an automatic process for obscuring private information and making Velvet robust to external attacks.

We evaluate the effectiveness of our approach through an experimental process to make Velvet more robust against external attacks aimed at prompting the LLM to generate memorized PII. We obtain Training Data Extraction (TDE) attacks from a subset of documents in Italian

CLiC-it 2025: Eleventh Italian Conference on Computational Linguistics, September 24 — 26, 2025, Cagliari, Italy
*Corresponding author.
g.xompero@almawave.it (G. A. Xompero)

[1]https://huggingface.co/Almawave/Velvet-2B

used to train Velvet to induce the model to leak PII; in particular, we focus on email addresses (Section 4.1). Then, we adapt PME to Velvet and edit the model to protect the LLM against identified TDE attacks (Section 4.2). Finally, we measure the effectiveness of our approach by observing the behaviour of Velvet against TDE attacks, and we evaluate the preservation of post-edit Velvet's multilingual generation capabilities to ensure the edit had no negative impact on the model (Section 4.3). Results show that model editing can be adapted to Italian data and make Velvet more robust against TDE attacks by notably reducing the accuracy of attacks (Section 5.1). In addition, evaluation of post-edit Velvet suggests that the edit does not affect multilingual capabilities for both English and Italian languages (Section 5.2).

2. Background

Given the large amount of data that is necessary to train an LLM, the risks connected to privacy violations have been largely investigated (Section 2.1). We describe what mechanisms in LLMs have been identified to control model predictions (Section 2.2), and how these insights allow editing some undesired predictions without the need of re-training the model (Section 2.3).

2.1. LLMs & Privacy

As LLMs require large amounts of data for training, some undesirable information may have been included in the training material inadvertently: a person's name, address, email address, social security number, phone number, as well as any other data that, when combined, could lead to identification of individuals, are considered private information and should not be further disseminated during inference by an LLM. This kind of information, defined as Personal Identifiable Information (PII), can in fact be used to identify a specific individual, and threats their privacy if disseminated.

However, once a PII is included in the training material, an LLM can leak it during inference. In fact, LLM may memorize that information [14, 15, 16] and consequently cause privacy leaks at inference time. A number of attacks have been designed to exploit this tendency and extract private information from LLMs [2, 17, 18]. For LLMs, even in black-box access the right prompt may be sufficient to obtain private information. While some attacks require the attacker to craft an adversarial input for the model [19, 20], other attacks do not even rely on potentially harmful prompts [3, 6, 4, 5].

Developing techniques for the preservation of individuals privacy is central to make LLMs more robust, and trustworthy.

2.2. Knowledge Mechanism of Transformers

Transformer-based Language Model Predictions We consider the forward pass of a Transformer-based decoder-only model $\mathcal{M}$ of L layers and describe it in terms of its sub-components on a prompt p. Given the tokenized prompt $X = [x_1, ..., x_n]$ and their corresponding input embeddings $X^{(0)}$, a model builds the prediction for the next token x_{n+1} with an iterative refinement across layers. At a given layer l, given the Attention Block as Attn, the layer normalization as LN and the Feed Forward block FFN, the output of that layer $X^{(l)}$ is computed as:

$$\forall l \in \{1, \dots, L\} : \begin{cases} A^{(l)} = \text{Attn}(\text{LN}(X^{(l-1)})) \\ \tilde{X}^{(l)} = X^{(l-1)} + A^{(l)} \\ H^{(l)} = \text{FFN}(\text{LN}(\tilde{X}^{(l)})) \\ X^{(l)} = \tilde{X}^{(l)} + H^{(l)} \end{cases} \tag{1}$$

On the last position n, at the last layer L, the hidden representation $x_n^{(L)}$ is projected by a matrix $U \in \mathbf{R}^{d \times |V|}$ onto the vocabulary V space. The scores obtained, normalized by a softmax function σ, predict the next token:

$$\mathcal{M}(X) = \arg\max \sigma \left(x_n^{(L)} U \right) = x_{n+1}$$

We aim to understand what are the mechanisms that control for the generation of next token, and if it is possible to alter them to modify the predictions on the next token when the model leaks private information.

FFN Layers as Knowledge Memories Feed Forward blocks FFN play a crucial role in the generation mechanism of the model, and not only because they account for most of the parameters of the network. The interpretation of the Feed-Forward block in a Transformer model is that it implements a mapping of paired keys to values [21, 22]. Geva et al. [21] notice that, with the exception of activation function that is usually a ReLU rather than a softmax, the equation for the Feed Forward layer reminds the one that describes a neural memory [23]. The Feed Forward block is in fact composed of two matrices, $W_{in}^{(l)}, {W_{out}^{(l)}}^T \in \mathbb{R}^{d \times d_1}$ and an activation function f that process each position $i \in [1, ..., n]$ of the input independently from one another. The output $h_n^{(l)}$ of the Feed Forward block at the n-th position of the input is computed as follow:

$$h_n^{(l)} = f \left(\tilde{x}^{(l)} W_{in}^{(l)} \right) W_{out}^{(l)} \tag{2}$$

where $\tilde{x}^{(l)}$ is the sum output of the Attention Block and the output of the previous layer as in Equation 1. The

keys of the memory are produced by the output of $W_{in}^{(l)}$ and the non-linear function f, while the values are the corresponding columns in $W_{out}^{(l)}$.

2.3. Editing Knowledge of LLMs

In the last years, there was a major interest around alternative methods to modify specific behaviors of LLMs without retraining the entire model from scratch. Based on the insights about the knowledge mechanism of transformers, the research area of knowledge editing has been flourishing, with the number of methods and approaches growing further.

Currently, knowledge editing methods can be roughly divided in two categories: parameter-preserving and parameter-editing methods [24]. While parameter-preserving methods rely on external adapters or memories to intervene whenever there is a specific situation requiring a different response, parameter-editing methods are based on the theory about the knowledge mechanism of transformers and modify the parameters of the LLM directly, without the need of external modules like parameter-preserving solutions.

We focus on parameter-editing methods: basically, given an LLM $\mathcal{M}_\theta$ with parameters θ, parameter-editing methods aim at finding a shift in parameters Δ to obtain a new model $M_{\theta+\Delta}$, which allows to modify a specific prediction while preserving the non-target generation capabilities. ROME [25] and MEMIT [26], in particular, are parameter-editing approaches designed to edit the LLMs' parameters in a localized manner and are based on the interpretation of Feed Forward layers as memories, as introduced in Section 2.2. Under this interpretation, then, the matrix $W_{out}^{(l)}$ is optimizing the mapping between keys and values, that is:

$$W_{out}^{(l)} = arg\min_{\widehat{W}} \sum_{(k_0,v_0)} ||\widehat{W}k_0 - v_0||^2 \tag{3}$$

with $k_0 \in K_0$ being a set of keys to memorize and $v_0 \in V_0$ the corresponding values [25, 26, 27]. Given the linearity of the system in Equation 3, the optimal solution can be computed as:

$$W_{out}^{(l)} = V_0 K_0^T (K_0 K_0^T)^{-1} \tag{4}$$

Additionally, a closed-form equation can be found to calculate the edit to introduce new keys and values into the mapping [25, 26]. Given a representation of keys K_0 and values V_0 stored in that matrix, and the representations for the new keys K^* and values V^* to store.

$$\Delta^{(l)} = (V^* - W_{out}^{(l)} K^*) K^{*T} (K_0 {K_0}^T + K^* K^{*T})^{-1} \tag{5}$$

The term $V^* - W_{out}^{(l)} K^*$ represents the residual between the new desired values V^* and the old values currently stored in $W_{out}^{(l)}$ for the new keys K^*. Since we have $K^* \subseteq K_0$ because the new keys are representations already stored in $W_{out}^{(l)}$, and the new values V_0^* satisfy $V_0^* \subseteq V_0$, we can define $W_{out}^{(l)} K^* = V_0^*$. The equation for $\Delta^{(l)}$ can be written as:

$$\Delta^{(l)} = (V^* - V_0^*) K^{*T} (K_0 {K_0}^T + K^* K^{*T})^{-1} \tag{6}$$

We will use the matrix $\Delta^{(l)}$ to edit the memorized mapping in layer l, without retraining.

Since we do not have access to K_0, Meng et al. [26] assumes that this representation can be modeled with a random sample of inputs, so $K_0 K_0^T$ can be defined as follows:

$$C_0^{(l)} = \lambda \cdot \mathbb{E}[kk^T] \triangleq K_0 {K_0}^T, \tag{7}$$

where $\lambda \cdot \mathbb{E}[kk^T]$ is an uncentered covariance statistics computed on an empirical sample of vector inputs to the layer. In this paper, we refer to it with C_0 rather than $C_0^{(l)}$ for simplicity.

2.4. Model Editing for Privacy Preservation

In recent studies, model editing techniques have been applied to the context of privacy protection.

Wu et al. [11] propose DEPN, which is a method that locates neurons associated with private information, and then edits their corresponding activations to remove their contribution to prediction.

Patil et al. [28] showed an application of ROME [25] and MEMIT [26] to remove private information from FFN layers of transformers. This approach exploits the association mechanism to break the associations leading to the leakage of private information.

Venditti et al. [12] propose PAE, a data-driven approach based on the editing mechanism of MEMIT, aiming to break the association between an individual and their corresponding PIIs. The method uses prompt templates filled with the information about an individual and their corresponding PII, to replace the private information with a dummy PII, thus preventing the leakage of the real PII.

Ruzzetti et al. [13] propose PME an automatic approach taking advantage of the memorization mechanism in LLMs. This approach basically uses memorized prompts inducing privacy violation to remove associated PIIs. Unlike other locate-and-edit methods, PME distributes the residual for the editing among all the FFN layers of the transformer. The main advantage of this method is that it can be used automatically on collected prompts without the need of further manual analysis to determine the source of the knowledge, allowing for an automatic algorithm for privacy protection.

In this paper, we apply PME because of its advantages, in particular the fact that it does not rely on assumptions such as which layers to modify or which part of a text retrieves the critical information, thus allowing for an automated process.

3. Application and Method

3.1. PII Leakage via Training Data Extraction attacks

PII is private information that may have been inadvertently included in the training dataset and can be extracted from an LLM using Training Data Extraction attacks (TDE) [3, 4, 5, 6]. In the initial formulation of TDE attacks, Carlini et al. [3] demonstrate that black-box access to an LLM can be sufficient to extract memorized information from a model: when prompted with a *context* that has been included in the training material, the target LLM tends to generate verbatim the continuation of the original document. Among the generated verbatim memorized content, a model may also generate private information that should not be disseminated.

Formally, given a model $\mathcal{M}$, a string s is k-extractable memorized if there exists a *context* string c of k tokens such that the concatenation of $[c \parallel s]$ is contained in the training material for $\mathcal{M}$ and $\mathcal{M}$ generates s exactly when prompted with c in greedy decoding. When the *context* exactly matches a sequence of the training material, the success of the attack is maximized [4], and since this is the most informative setting that the attacker can obtain, this is the worst-case scenario.

The success of the attack increases as the attacker gets more information regarding the training material: one crucial aspect is the length k of the *context* that the model is fed with [5, 4]: the longer the *context*, the larger the probability of emission of verbatim memorized information.

Since LLMs have been shown to memorize PII rather than associating them with an individual identity [5, 12, 2], those attacks represent one of the crucial challenges to protect individuals whom information have been inadvertently added to the training material of an LLM.

Hence, we initially perform TDE attacks against our target model: we simulate an informed attacker who has some background knowledge regarding the training material, with increasing level of information. For a given PII, we collect the *context* that precede it in the training materials, and produce 50, 100, and 200-tokens-long sequences (see Section 4.1 for further details) as we expect that a more informed attacker may obtain larger volume of information. The model is then prompted to generate the subsequent 100 tokens: the attack succeeds if – in greedy decoding – the generated PII matches the original PII in the training material: the evaluation is rigorous since a strict match between the generated PII and the one found in the training material is required.

3.2. PME for Automatic Privacy Mitigation

To address the threats posed by TDE attacks, we adopt Private Memorization Editing (PME) [13], a model editing strategy that aims to leverage the memorization tendencies of LLMs as a defense strategy. The objective of the method is to reduce the success of TDE attacks, and hence to replace the memorized PII with a semantically equivalent, but privacy-preserving information. PME applies the editing on the Feed Forward layers of the models, similarly to other model editing techniques like ROME [25] and MEMIT [26].

As discussed in Section 2.3, once one knows the correct representation for keys and values that the $W_{out}^{(l)}$ encodes, it is possible to apply the closed form solution in Equation 6 to perform the update. To compute the correct representation for keys and values, PME directly exploits training material verbatim memorized from a model.

When the model is prompted with a context c that is included in the training material that causes the generation of a PII, PME edits the model to obtain a privacy-preserving output instead. In each layer, the keys are the hidden representation that the model computes for the *context* prompt as in Equation 2, so $k^{(l)} = f\left(\tilde{x}^{(l)} W_{in}^{(l)}\right)$.

For the values, the new privacy-preserving value should be encoded with an appropriate vector representation. For this reason, PME initially optimizes a hidden representation v^* in the last layer of the model: using Gradient Descent, PME optimizes v^* so that, once decoded with the projection matrix on the vocabulary, it gives the highest probability of generating a dummy, privacy-preserving value.

Then, the underlying hypothesis in PME is that each layer should contribute to the generation of this last-layer representation v^*. PME mimics the generation of the PII: with a forward pass on the memorized context, the method quantifies how much each layer contributes to the generation of the memorized PII. Instead of relying on Causal Mediation Analysis as in MEMIT [26] or other localization techniques that have been shown to not inform the edit [29, 30] for identifying a restricted number of contributing layers, a *contribution coefficient* is computed for each layer following a geometric approach. Since the computation of a Transformer model can be described as a sum of its sub-components at each layer [31, 32], PME computes the *contribution coefficient* $w^{(l)}$ of each layer as the projection of that layer Feed Forward output onto the last layer Feed Forward representation:

the larger the projection, the larger the impact of that layer on the overall sum. This *contribution coefficient* – rescaled to obtain a sum of one across different layers – is then used to represent a fraction of v^*, proportionally to the *contribution coefficient* $w^{(l)}$ of that layer, that is, at each layer the value $v^{(l)} = w^{(l)}v^*$

Once the correct representation for keys and privacy preserving values is computed, then the edit can be performed as in Equation 6, and the post-edit model should not generate the target PII under TDE attacks.

4. Experimental Setting

In this section, we discuss the experimental setting we use to assess the effectiveness of our approach. Specifically, we define: (1) the process for data preparation to obtain the TDE attacks and relative leaked information (Section 4.1), (2) how PME is adapted and applied to Velvet (Section 4.2), and (3) how we evaluate the effectiveness of our privacy protection approach and the post-edit preservation of Velvet's capabilities (Section 4.3). For these experiments, we focus on email addresses of Italian data as PII, and Velvet-2B as our target LLM.

4.1. Data Preparation

Training Data Extraction Attacks As we discussed in Section 3.1, Training Data Extraction attacks are based on documents and prompts that the LLM has seen during training, which induce a target LLM to complete the given prompts with a text verbatim memorized by the model. Since LLMs are prone to leak PII during generation due to possible contamination of training data with PII, we prepare Training Data Extraction attacks by analyzing a subset of the training data used for Velvet. We focus on the Italian subset of *CulturaY* [33], one of the public datasets seen by Velvet during the pre-training phase.

We focus on potentially harmful prompts, since our main objective is to study the feasibility of protecting against TDE rather than assessing their accuracy. To do that, we define the following protocol. We filter all documents in the dataset that contain at least one email address in them. Then, once we obtain only documents containing PII, we prepare batches of different potential TDE attack prompts of different lengths $k \in \{50, 100, 200\}$, by selecting the k tokens preceding the identified PII. After obtaining a set of potential attacks, we deduplicate similar prompts. In order to select effective attacks, we prompt Velvet-2B with the collected attacks and induce the model to generate 100 tokens: if the email address generated by the model for a given prompt is the one expected as in the training data, we add it to the set of TDE attacks.

Sample for computing PME Editing Statistics An important step required by PME to perform the desired edit is the uncentered covariance statistic $C_0^{(l)}$ described in Eq.7. This is an estimation of the keys stored in the corresponding l-th FFN layer, so we need to build an empirical sample of vector inputs for the layer, which are obtained by feeding the LLM with sample texts. Since we are dealing with a multilingual LLM trained on both English and Italian texts, we prepare two samples of 100k documents each from English and Italian Wikipedia subsets of the pre-training data used for Velvet-2B. The purpose of these samples is to understand the effects on the editing performance of C_0 computed on different languages.

4.2. Application of PME

Mitigating Privacy Leakage Our strategy is to prevent Velvet from generating memorized PIIs during inference by applying PME to Velvet on identified TDE attacks reported in Section 4.1. PME allows to edit the relative knowledge of PII associated with multiple memorized prompts by modifying the LLM's parameters directly. The main advantage of this method is that we can edit the TDE attacks directly and there is no need to specify which layers are the target of the edit, unlike methods such as MEMIT [26].

Based on this, for every attack (x, y) with $y = \mathcal{M}(x)$, x the prompt attack and y the leaked PII, we use PME to edit the knowledge encoded in Velvet's FFN layers to force the new association (x, z), where z is the new dummy PII `mail@domain.com`, which is semantically similar to the original PII. With this method, our objective is to reduce the accuracy of attacks, modifying the prediction of the LLM to prevent the generation of the leaked information.

We perform the editing process with an approach called sequential batch editing [12, 13], in which several prompts are edited in multiple steps, with a batch of multiple examples edited at each step. For our experiments, we fixed the batch size to 16.

Computing Multilingual C_0 for PME PME [13], ROME [25] and MEMIT [26] require a representation of the keys K_0 stored in the l-th FFN layer to apply the formula defined in Eq.6, which can be modeled as the quantity $C_0^{(l)}$ defined in Eq.7. This quantity is obtained by computing an uncentered covariance statistics on an empirical sample of vector inputs to the layer when parsing a sample of documents. For our experiments, we prepare three types of C_0 for PME on the text samples described in Section 4.1:

- `IT`: computed on the Italian sample;
- `EN`: computed on the English sample;

Velvet-2B	PME C_0	TDE Email Attacks				
		Context Length	**Tot. Prompts**	**Generated PII**	**Leaked PII**	**Attack Acc.**
Pre-edit	-	50	83	78	75	0.904
Post-Edit	multi		83	51	7	**0.084**
	EN		83	51	8	0.096
	IT		83	46	9	0.108
Pre-Edit	-	100	380	370	341	0.897
Post-Edit	multi		380	151	16	0.042
	EN		380	128	12	**0.032**
	IT		380	157	15	0.039
Pre-Edit	-	200	34	32	31	0.912
Post-Edit	multi		34	25	16	**0.471**
	EN		34	27	17	0.5
	IT		34	27	16	**0.471**

Table 1
Attack Accuracy results for TDE Email attacks on Pre-edit and Post-edit versions of Velvet-2B. We report the number of attacks **Tot. Prompts** of length **Context Length**, the number of generic email addresses generated by the models **Generated PII**, the quantity of email addresses leaked **Leaked PII**, and the C_0 type used for editing **PME** C_0.

- `multi`: computed on the English and Italian samples combined.

We compute these statistics for all the FFN layers of Velvet following the same procedure carried out by Meng et al. [26].

This statistic plays a crucial role in Eq.6, as it allows us to determine the interaction between the new keys and the knowledge stored in that layer. An effective computation of this statistic is necessary to obtain effective edits, and we empirically explore how different estimates of C_0 may affect the edit in a multilingual setting.

4.3. Evaluation

Post-Edit Attack Accuracy PME effectively protects the privacy in Velvet if the parameter edit reduces the number of successful TDE attacks against the model. Therefore, the effectiveness of our approach is assessed by measuring the post-edit privacy leakage effects and comparing them with the ones of the pre-edit model.

We adopted the same measure used by Ruzzetti et al. [13], that is, the *Attack Accuracy* for memorization attacks. After we edit Velvet for TDE attacks of $k \in \{50, 100, 200\}$ context lengths, we measure the *Attack Accuracy* of post-edit models and compare their scores with the ones of the pre-edit version of Velvet. We feed the TDE prompts to both post-edit and pre-edit version of Velvet, and then let them generate 100 tokens: if the generated text for each attack contains the expected PII, then the attack is considered successful.

Post-Edit Multilingual Generation Capabilities An important aspect of model editing methods is that they are designed to modify specific knowledge of LLMs, while preserving the non-related generative capabilities of the model. For this reason, we need to determine whether the editing had a negative impact on the multilingual generative capabilities of our LLM, thus affecting its skills in non-related tasks.

We adopt an automatic evaluation strategy similar to the one used by Venditti et al. [12] to measure the reliability of our post-edit models. We compare the generation capabilities of the post-edit and pre-edit versions of Velvet by measuring the similarity of generated texts on a sample of prompts in terms of BLUE [34] and METEOR [35] scores. For comparison, we consider the subsequent 50 tokens generated by each model after receiving in input the first 100 token of each prompt of our sample.

We perform the evaluation on a sample of 500 prompts for the English and Italian languages, which is defined as follows:

- English sample: 100 prompts from Books3, Wikipedia-en, and Pile-CC subsets of the Pile, respectively;
- Italian sample: 100 prompts from Clean-C4 and Wikipedia-it, respectively.

The composition of this sample allows to have an indication of the impact of PME editing on post-edit language capabilities of Velvet.

We also extend the *utility* evaluation by measuring the post-edit accuracy of Velvet on LAMBADA[36], one of the tasks included in EleutherAI Language Model Evaluation Harness[37]. LAMBADA is used to measure the accuracy of a model in generating the missing target word from a passage given in input. For the evaluation, we focus on the full test split of the dataset to measure the reliability of the edit. Since we are interested in evaluating the preservation of the post-edit multilingual capabilities of the model, we use both the English and Italian

PME C_0	Editing Attacks		Books3 (EN)		Wikipedia (EN)		Pile-CC (EN)		Clean C4 (IT)		Wikipedia (IT)	
	Context	*Prompts*	*BLEU*	*METEOR*	*BLEU*	*METEOR*	*BLEU*	*METEOR*	*BLEU*	*METEOR*	*BLEU*	*METEOR*
IT	50	83	**84.4 (±11.2)**	87.3 (±11.2)	88.6 (±12.4)	91.1 (±10.7)	86.1 (±11.6)	89.5 (±9.4)	**86.0 (±11.1)**	**89.8 (±9.1)**	90.3 (±13.7)	**93.2 (±10.0)**
EN			83.6 (±11.1)	87.0 (±10.9)	**90.2 (±11.6)**	**92.1 (±9.8)**	86.2 (±12.0)	89.6 (±10.0)	82.2 (±10.4)	87.1 (±9.3)	88.1 (±13.2)	92.3 (±9.9)
multi			84.3 (±11.4)	**87.5 (±11.0)**	89.3 (±12.2)	91.3 (±10.4)	**86.6 (±11.6)**	**89.9 (±9.9)**	**86.0 (±10.8)**	89.3 (±9.4)	**91.1 (±12.5)**	93.0 (±10.6)
IT	100	380	82.7 (±11.4)	86.2 (±10.9)	86.1 (±14.2)	89.2 (±12.1)	84.5 (±11.8)	88.9 (±9.8)	84.4 (±10.2)	88.0 (±9.4)	**90.0 (±13.1)**	**93.3 (±9.3)**
EN			**84.6 (±11.0)**	**87.5 (±10.8)**	**88.8 (±12.4)**	**91.1 (±10.7)**	85.6 (±11.8)	89.0 (±9.6)	81.1 (±10.7)	85.9 (±9.7)	87.3 (±13.7)	91.4 (±10.3)
multi			84.4 (±11.0)	86.8 (±11.1)	87.7 (±14.1)	90.7 (±11.3)	**86.2 (±11.3)**	**89.3 (±9.7)**	**84.6 (±10.7)**	**88.4 (±9.5)**	89.3 (±13.2)	92.7 (±9.4)
IT	200	34	83.9 (±10.8)	86.9 (±10.5)	88.6 (±12.3)	**91.5 (±10.2)**	86.1 (±11.1)	89.1 (±10.1)	**86.7 (±11.4)**	**89.8 (±10.1)**	**90.9 (±13.0)**	**93.6 (±9.9)**
EN			84.8 (±11.9)	**88.0 (±10.8)**	88.2 (±12.9)	90.5 (±11.2)	**87.1 (±10.8)**	89.7 (±9.7)	85.3 (±10.8)	88.4 (±9.8)	90.0 (±13.0)	93.5 (±9.3)
multi			**85.0 (±11.2)**	87.5 (±10.8)	**89.5 (±11.9)**	**91.5 (±10.2)**	87.0 (±11.0)	**89.9 (±9.8)**	85.7 (±10.9)	88.4 (±9.8)	89.7 (±15.0)	92.5 (±12.6)

Table 2
Post-edit Automatic Evaluation on English and Italian text samples, compared with the pre-edit generations. **PME** C_0 is the type of C_0 applied to edit the model, and **Editing Attacks** are the prompts used by PME to remove private information, with *Context* the length of TDE prompts *Prompts*.

versions of LAMBADA to understand if the multilingual generation capabilities of Velvet have been affected.

5. Results and Discussion

5.1. Editing reduces Privacy Risks

As we observed during the extraction and filtering phase of TDE attacks (see Sec. 4.1), Velvet memorized some PII contained in the pre-training data. For different context lengths $k \in \{50, 100, 200\}$, we obtained 83, 380, and 34 leaked email addresses, respectively, with the same number of memorized prompts. Surprisingly, context of 200 tokens obtained less leaked PII than shorter prompts. In this phase, we observe that a slightly different prompt composition might affect the results: so in pre and post-edit we adopt the same batch size and batch composition, to ensure the reproducibility of the results.

The results reported in Table 1 show that PME is effective in reducing the risks of privacy leakage. The post-edit versions of Velvet for contexts 50 and 100 are more robust than the pre-edit model, leaking less than 9 and 16 PII with respect to 75 and 341 leaked by the pre-edit Velvet. The effect is similar for all the versions of C_0 used by PME for editing, with minimal differences among them: in fact, the difference is of 4 more leaked PII at best for context 100.

The number of leaked email addresses is reduced even for context 200 attacks, where post-edit Velvet leaked 17 PII instead of 31 of the pre-edit model. However, the reduction here is lower compared with the other attacks, probably due to the lower number of PII extracted during the data processing phase.

Note that results also show that the model tends to generate a large number of email addresses in general, which are different from the correct ones. These different email addresses could be model's hallucinations, or email addresses that follow the original one in the pre-training corpus. However, results in terms of successfully Leaked PII suggest that PME is still sufficiently effective in preserving privacy on edited prompts.

Finally, we observe that the different statistics computed as an approximation of C_0 do not greatly affect the post-edit attack accuracy, with a rather similar number of leaked PII in each configuration.

5.2. Generation Capabilities are Preserved

The results reported in Table 2 show that BLEU and METEOR scores are high in general for all the different versions of C_0 and attacks used for editing, and the same observation holds for both English and Italian generation capabilities. The overall high scores suggest that the generations of post-edit models are quite similar to the generated texts of the pre-edit model. This aspect, as discussed in [12], suggests that the edit is robust, because it does not interfere with multilingual capabilities in both English and Italian languages.

Interestingly, the scores show that there is no real consensus on the type of statistics that is the best for the English language, since the highest scores are shared between the EN and `multi` C_0. However, we note that the IT version of C_0 obtains lower scores than the other two versions in general, suggesting that the IT statistics leads to a less effective preservation of Velvet's generation capabilities for English.

Observing the evaluation results for Italian, we notice that IT version of C_0 achieves higher BLEU and METEOR scores, suggesting that this version is necessary to preserve the generation capabilities of Velvet for Italian. Also, we note that the EN version of C_0 tends to achieve lower scores with respect to the other types, indicating that this C_0 is less effective for preserving the abilities for Italian.

In general, observed results indicate that using versions of C_0 computed on a different language from the target one is less effective for preserving the generative capabilities of the target language in post-edit. In fact, the IT version of C_0 obtained lower scores for the English language, and the EN version of C_0 was less effective for the Italian language. Thus, these experiments suggest that C_0 should be computed on samples containing texts in the target languages.

Velvet-2B	PME C_0	Editing Attacks		LAMBADA	
		Context	*Prompts*	*EN*	*IT*
Pre-edit	-	-		53.7	45.2
Post-Edit	multi	50	83	54.2	45.1
	EN			53.9	45.2
	IT			54.4	45.0
Post-Edit	multi	100	380	54.5	45.2
	EN			54.7	42.1
	IT			55.1	45.2
Post-Edit	multi	200	34	54.1	45.0
	EN			53.9	45.9
	IT			54.1	45.2

Table 3
LAMBADA scores for the pre-edit and post-edit versions of Velvet-2B. Results for both English and Italian are comparable with the pre-edit model, suggesting that capabilities of Velvet-2B are preserved in post-edit.

About task performance, results reported in Table 2 of the LAMBADA benchmark corroborate the utility preservation already observed with the previous evaluation analysis. The accuracy scores of post-edit models are comparable with the pre-edit ones, suggesting that the edits performed by PME do not affect considerably the capabilities of the model. The same observation holds for both English and Italian versions of LAMBADA. Differently from the previous analysis, there are no noticeable losses in terms of performance with respect to the version of C_0 used for the editing, except for the Italian score of context-100 editing with EN C_0 that is lower than the pre-edit score (42.1 vs 45.2). Hence, this result indicates that edits performed by PME are reliable in general, allowing privacy protection of Velvet for Italian data without loss of task performance.

6. Conclusions and Future Work

In this work, we show an application of model editing for protecting the privacy of Italian data on Velvet-2B, a multilingual model trained on both Italian and English data.

Our method is based on a recent model editing technique named Private Memorization Editing, which prevents LLMs from generating memorized PII that might be included in the training data. Results of our experiments on privacy protection for email addresses shows that model editing is effective in reducing the privacy risks of Velvet, thus reducing the success of Training Data Extraction (TDE) attacks, harmful prompts obtained from the training data that are effective for extracting private information from the original model. In addition, we show that our approach mitigates the privacy risks while preserving the model's multilingual generation capabilities.

In conclusion, our approach shows that we can adapt and apply model editing techniques for privacy protection in multilingual LLMs for Italian data.

For future work, we should focus on some other aspects to further improve this work. Firstly, our approach should be extended to different types of PII other than email addresses, and further investigation is necessary to understand the effects of the approach with different PII. Another aspect to consider is how well PME scales with larger models such as Velvet-14B: this other model requires additional investigation, because it manages other languages other than English and Italian, and the magnitude of data used for training is larger than the one used for Velvet-2B. Finally, the evaluation of Velvet's post-edit capabilities should be extended to other tasks of the Language Model Evaluation Harness[37] or other benchmarks, and include human evaluation to have a better perspective on the overall quality of post-edit models instead of relying exclusively on automatic metrics.

References

[1] R. Orlando, L. Moroni, P.-L. Huguet Cabot, S. Conia, E. Barba, S. Orlandini, G. Fiameni, R. Navigli, Minerva LLMs: The first family of large language models trained from scratch on Italian data, in: F. Dell'Orletta, A. Lenci, S. Montemagni, R. Sprugnoli (Eds.), Proceedings of the 10th Italian Conference on Computational Linguistics (CLiC-it 2024), CEUR Workshop Proceedings, Pisa, Italy, 2024, pp. 707–719. URL: https://aclanthology.org/2024.clicit-1.77/.

[2] M. Miranda, E. S. Ruzzetti, A. Santilli, F. M. Zanzotto, S. Bratières, E. Rodolà, Preserving privacy in large language models: A survey on current threats and solutions, Transactions on Machine Learning Research (2025). URL: https://openreview.net/forum?id=Ss9MTTN7OL.

[3] N. Carlini, F. Tramer, E. Wallace, M. Jagielski, A. Herbert-Voss, K. Lee, A. Roberts, T. Brown, D. Song, U. Erlingsson, et al., Extracting training data from large language models, in: 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 2633–2650.

[4] N. Carlini, D. Ippolito, M. Jagielski, K. Lee, F. Tramer, C. Zhang, Quantifying memorization across neural language models, 2023. arXiv:2202.07646.

[5] J. Huang, H. Shao, K. C.-C. Chang, Are large pre-trained language models leaking your personal information?, in: Y. Goldberg, Z. Kozareva, Y. Zhang (Eds.), Findings of the Association for Computational Linguistics: EMNLP 2022, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 2038–2047. URL: https://

aclanthology.org/2022.findings-emnlp.148. doi:10.18653/v1/2022.findings-emnlp.148.

[6] M. Nasr, N. Carlini, J. Hayase, M. Jagielski, A. F. Cooper, D. Ippolito, C. A. Choquette-Choo, E. Wallace, F. Tramèr, K. Lee, Scalable extraction of training data from (production) language models, arXiv preprint arXiv:2311.17035 (2023).

[7] T. Nguyen, C. V. Nguyen, V. D. Lai, H. Man, N. T. Ngo, F. Dernoncourt, R. A. Rossi, T. H. Nguyen, CulturaX: A cleaned, enormous, and multilingual dataset for large language models in 167 languages, in: N. Calzolari, M.-Y. Kan, V. Hoste, A. Lenci, S. Sakti, N. Xue (Eds.), Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024), ELRA and ICCL, Torino, Italia, 2024, pp. 4226–4237. URL: https://aclanthology.org/2024.lrec-main.377.

[8] L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, S. Presser, C. Leahy, The pile: An 800gb dataset of diverse text for language modeling, 2020. arXiv:2101.00027.

[9] Y. Yao, X. Xu, Y. Liu, Large language model unlearning, 2024. URL: https://arxiv.org/abs/2310.10683. arXiv:2310.10683.

[10] A. Kassem, O. Mahmoud, S. Saad, Preserving privacy through dememorization: An unlearning technique for mitigating memorization risks in language models, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 4360–4379. URL: https://aclanthology.org/2023.emnlp-main.265. doi:10.18653/v1/2023.emnlp-main.265.

[11] X. Wu, J. Li, M. Xu, W. Dong, S. Wu, C. Bian, D. Xiong, DEPN: Detecting and editing privacy neurons in pretrained language models, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Singapore, 2023, pp. 2875–2886. URL: https://aclanthology.org/2023.emnlp-main.174. doi:10.18653/v1/2023.emnlp-main.174.

[12] D. Venditti, E. S. Ruzzetti, G. A. Xompero, C. Giannone, A. Favalli, R. Romagnoli, F. M. Zanzotto, Enhancing data privacy in large language models through private association editing, 2024. URL: https://arxiv.org/abs/2406.18221. arXiv:2406.18221.

[13] E. S. Ruzzetti, G. A. Xompero, D. Venditti, F. M. Zanzotto, Private memorization editing: Turning memorization into a defense to strengthen data privacy in large language models, in: W. Che, J. Nabende, E. Shutova, M. T. Pilehvar (Eds.), Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Vienna, Austria, 2025, pp. 16572–16592. URL: https://aclanthology.org/2025.acl-long.810/.

[14] S. Biderman, U. Prashanth, L. Sutawika, H. Schoelkopf, Q. Anthony, S. Purohit, E. Raff, Emergent and predictable memorization in large language models, Advances in Neural Information Processing Systems 36 (2023) 28072–28090.

[15] F. Ranaldi, E. S. Ruzzetti, D. Onorati, L. Ranaldi, C. Giannone, A. Favalli, R. Romagnoli, F. M. Zanzotto, Investigating the impact of data contamination of large language models in text-to-SQL translation, in: L.-W. Ku, A. Martins, V. Srikumar (Eds.), Findings of the Association for Computational Linguistics: ACL 2024, Association for Computational Linguistics, Bangkok, Thailand, 2024, pp. 13909–13920. URL: https://aclanthology.org/2024.findings-acl.827/. doi:10.18653/v1/2024.findings-acl.827.

[16] H. Kiyomaru, I. Sugiura, D. Kawahara, S. Kurohashi, A comprehensive analysis of memorization in large language models, in: S. Mahamood, N. L. Minh, D. Ippolito (Eds.), Proceedings of the 17th International Natural Language Generation Conference, Association for Computational Linguistics, Tokyo, Japan, 2024, pp. 584–596. URL: https://aclanthology.org/2024.inlg-main.45/.

[17] B. Yan, K. Li, M. Xu, Y. Dong, Y. Zhang, Z. Ren, X. Cheng, On protecting the data privacy of large language models (llms): A survey, arXiv preprint arXiv:2403.05156 (2024).

[18] A. Verma, S. Krishna, S. Gehrmann, M. Seshadri, A. Pradhan, T. Ault, L. Barrett, D. Rabinowitz, J. Doucette, N. Phan, Operationalizing a threat model for red-teaming large language models (llms), arXiv preprint arXiv:2407.14937 (2024).

[19] F. Perez, I. Ribeiro, Ignore previous prompt: Attack techniques for language models, in: NeurIPS ML Safety Workshop, 2022.

[20] X. Shen, Z. Chen, M. Backes, Y. Shen, Y. Zhang, " do anything now": Characterizing and evaluating in-the-wild jailbreak prompts on large language models, in: Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, 2024, pp. 1671–1685.

[21] M. Geva, R. Schuster, J. Berant, O. Levy, Transformer feed-forward layers are key-value memories, in: M.-F. Moens, X. Huang, L. Specia, S. W.-t. Yih (Eds.), Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Online and Punta Cana, Domini-

can Republic, 2021, pp. 5484–5495. URL: https://aclanthology.org/2021.emnlp-main.446. doi:10.18653/v1/2021.emnlp-main.446.

[22] M. Geva, A. Caciularu, K. Wang, Y. Goldberg, Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space, in: Y. Goldberg, Z. Kozareva, Y. Zhang (Eds.), Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 30–45. URL: https://aclanthology.org/2022.emnlp-main.3. doi:10.18653/v1/2022.emnlp-main.3.

[23] S. Sukhbaatar, J. Weston, R. Fergus, et al., End-to-end memory networks, Advances in Neural Information Processing Systems 28 (2015).

[24] Y. Yao, P. Wang, B. Tian, S. Cheng, Z. Li, S. Deng, H. Chen, N. Zhang, Editing large language models: Problems, methods, and opportunities, 2023. arXiv:2305.13172.

[25] K. Meng, D. Bau, A. Andonian, Y. Belinkov, Locating and editing factual associations in gpt, 2023. arXiv:2202.05262.

[26] K. Meng, A. S. Sharma, A. Andonian, Y. Belinkov, D. Bau, Mass-editing memory in a transformer, 2023. arXiv:2210.07229.

[27] T. Kohonen, Correlation matrix memories, IEEE Transactions on Computers C-21 (1972) 353–359. URL: https://api.semanticscholar.org/CorpusID:21483100.

[28] V. Patil, P. Hase, M. Bansal, Can sensitive information be deleted from llms? objectives for defending against extraction attacks, 2023. arXiv:2309.17410.

[29] T.-Y. Chang, J. Thomason, R. Jia, Do localization methods actually localize memorized data in LLMs? a tale of two benchmarks, in: K. Duh, H. Gomez, S. Bethard (Eds.), Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 3190–3211. URL: https://aclanthology.org/2024.naacl-long.176/. doi:10.18653/v1/2024.naacl-long.176.

[30] P. Hase, M. Bansal, B. Kim, A. Ghandeharioun, Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models, 2023. URL: https://arxiv.org/abs/2301.04213. arXiv:2301.04213.

[31] T. Mickus, D. Paperno, M. Constant, How to dissect a Muppet: The structure of transformer embedding spaces, Transactions of the Association for Computational Linguistics 10 (2022) 981–996. URL: https://aclanthology.org/2022.tacl-1.57. doi:10.1162/tacl_a_00501.

[32] J. Ferrando, G. Sarti, A. Bisazza, M. R. Costa-jussà, A primer on the inner workings of transformer-based language models, 2024. URL: https://arxiv.org/abs/2405.00208. arXiv:2405.00208.

[33] H. N. Thuat Nguyen, T. Nguyen, Culturay: A large cleaned multilingual dataset of 75 languages, 2024.

[34] T. Glushkova, C. Zerva, A. F. T. Martins, BLEU meets COMET: Combining lexical and neural metrics towards robust machine translation evaluation, in: M. Nurminen, J. Brenner, M. Koponen, S. Latomaa, M. Mikhailov, F. Schierl, T. Ranasinghe, E. Vanmassenhove, S. A. Vidal, N. Aranberri, M. Nunziatini, C. P. Escartín, M. Forcada, M. Popovic, C. Scarton, H. Moniz (Eds.), Proceedings of the 24th Annual Conference of the European Association for Machine Translation, European Association for Machine Translation, Tampere, Finland, 2023, pp. 47–58. URL: https://aclanthology.org/2023.eamt-1.6.

[35] S. Banerjee, A. Lavie, METEOR: An automatic metric for MT evaluation with improved correlation with human judgments, in: J. Goldstein, A. Lavie, C.-Y. Lin, C. Voss (Eds.), Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, Association for Computational Linguistics, Ann Arbor, Michigan, 2005, pp. 65–72. URL: https://aclanthology.org/W05-0909.

[36] D. Paperno, G. Kruszewski, A. Lazaridou, N. Q. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, R. Fernández, The LAMBADA dataset: Word prediction requiring a broad discourse context, in: K. Erk, N. A. Smith (Eds.), Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Berlin, Germany, 2016, pp. 1525–1534. URL: https://aclanthology.org/P16-1144. doi:10.18653/v1/P16-1144.

[37] L. Gao, J. Tow, B. Abbasi, S. Biderman, S. Black, A. DiPofi, C. Foster, L. Golding, J. Hsu, A. Le Noac'h, H. Li, K. McDonell, N. Muennighoff, C. Ociepa, J. Phang, L. Reynolds, H. Schoelkopf, A. Skowron, L. Sutawika, E. Tang, A. Thite, B. Wang, K. Wang, A. Zou, A framework for few-shot language model evaluation, 2024. URL: https://zenodo.org/records/12608602. doi:10.5281/zenodo.12608602.