PrivacyRestore: Privacy-Preserving Inference in Large Language Models via Privacy Removal and Restoration

Ziqian Zeng* [1], **Jianwei Wang*** [1], **Junyao Yang*** [1],
Zhengdong Lu[1], **Haoran Li**[3], **Huiping Zhuang** † [1], **Cen Chen** † [1,2]
[1]South China University of Technology
[2] Pazhou Laboratory, China
[3] Hong Kong University of Science and Technology
zqzeng@scut.edu.cn, wiwjwilliam@mail.scut.edu.cn

Abstract

The widespread usage of online Large Language Models (LLMs) inference services has raised significant privacy concerns about the potential exposure of private information in user inputs. Existing privacy protection methods for LLMs suffer from either insufficient privacy protection with performance degradation, or large inference time overhead. To address these limitations, we propose PrivacyRestore, a plug-and-play method to protect the privacy of user inputs during LLM inference for the client-server scenario. The server first trains restoration vectors for each privacy span type offline and then releases them to the clients. During inference, the client aggregates restoration vectors of all privacy spans in the user query into a meta restoration vector which is later sent to the server to restore information. Before transmission, the client removes all privacy spans in the user query and applies d_χ-privacy mechanism to the meta vector for privacy protection. We prove that our method can inherently prevent the linear growth of the privacy budget. We conduct extensive experiments, covering the medical and legal domains, and demonstrate that PrivacyRestore effectively protects private information and maintains acceptable levels of performance and inference efficiency [1] .

1 Introduction

Large language models (LLMs) have emerged as powerful tools across various domains (Chen et al., 2023b; Wu et al., 2023). However, the widespread use of online LLM inference services has raised significant privacy concerns. When interacting with LLMs deployed on cloud platforms, users' inputs may contain sensitive data, such as medical records and legal case details. Potential threats may arise when attackers intercept user queries during data transmission, and some advanced adversaries can even hack the cloud service provider. For example, in sensitive domains like medical diagnosis, if a user's input containing the user's personal protected health information (PHI), such as "*I was previously diagnosed with HIV, and lately I've been experiencing fever and diarrhea...*" is disclosed to malicious attackers, it may cause privacy concerns.

In this paper, we focus on protecting the private information contained in user inputs during LLM inference. In this setting, the client submits inputs to the server (also known as the service provider) and there is a risk that inputs might be disclosed by attackers. Current methods for protecting user inputs can be mainly divided into two categories: Secure Multi-Party Computation (SMPC) and Differential Privacy (DP). SMPC-based methods (Hao et al., 2022a; Li et al., 2023a; Liang et al., 2024) utilize encryption protocols and algorithms to enable collaborative computation without revealing original data to others. However, SMPC methods have large inference time overheads, making them impractical for real-time applications (Hao et al., 2022b). DP based methods (Feyisetan et al., 2020, 2019; Xu et al., 2020; Bo et al., 2021) apply d_χ-privacy (Chatzikokolakis et al., 2013; Alvim et al., 2018) to words and achieve word-level text-to-text privatization. Nevertheless, DP-based methods inevitably degrade the performance of downstream tasks due to noise injection, which is known as the privacy-utility trade-off. Additionally, as the text length grows, word-level privatization will lead to significant performance degradation. This phenomenon is known as the linear growth of the privacy budget in word-level privatization (Mattern et al., 2022b). Hence, there is a need to develop privacy-preserving methods which can effectively safeguard the privacy of user inputs while maintaining high-quality outputs, without incurring prohibitive computational costs.

We propose PrivacyRestore, which directly re-

*Equal contribution.
†Corresponding author
[1]We release our code and datasets in https://github.com/ZeroNLP/PrivacyRestore

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10821–10855
July 27 - August 1, 2025

moves privacy spans in user inputs and restores private information via activation steering (Li et al., 2023c; Turner et al., 2023; Hernandez et al., 2023) during model inference. Our method is based on two key phenomena: **(a) Users' private information mostly consists of sensitive attributes and these attributes are commonly confined within specific contiguous token sequences, referred to as "privacy spans".** For eaxmple, in the context of healthcare domain, the private information may commonly refer to symptom descriptions. Consider a medical record which states "*I was previously diagnosed with HIV, and lately I've been experiencing fever and diarrhea...*", "*HIV*" and "*fever and diarrhea*" should be protected as privacy spans. Directly removing these privacy spans (symptom descriptions) can significantly hinder attackers from reconstructing or inferring private information and serves as an effective approach to preventing privacy leakage. **(b) Most privacy spans are concentrated in a few majority categories, exhibiting a long-tailed distribution.** For instance, in medical diagnosis applications, privacy spans typically relate to symptoms and disease descriptions. Most of the symptoms and disease descriptions appearing in user inputs are concentrated on high-frequency types, such as "*fever*" and "*cold*". We have conducted experiments to demonstrate the long-tailed distribution of privacy spans, as detailed in Appendix K.

PrivacyRestore operates in two stages: the preparation stage and the inference stage. **In the preparation stage**, the server first identifies the attention heads where the activation steering occurs. Second, each privacy span type is encoded to a vector, known as the restoration vector. This stage is performed entirely offline on the server side. Our method is plug-and-play, requiring only the restoration vectors to be trainable, while keeping the LLM frozen. Once training is complete, these restoration vectors will be released to the client side. **In the inference stage**, according to the principle of "Information Self-Determination Right" [2] (Jasper M C, 2009; Alsenoy et al., 2014) , the users are entitled to identify the privacy spans in their inputs by themselves. After identification, a meta vector is constructed by estimating the importance of each privacy span and calculating a weighted sum of the corresponding restoration vectors. Then, the user submits the incomplete input with the privacy spans removed, along with the meta vector, to the server. The server uses the meta vector to restore the removed privacy spans and generate high-quality outputs.

To prevent the leakage of privacy spans via the meta vector, d_χ-privacy mechanisms are applied to the meta vector before transmission at the client side. By applying d_χ-privacy to the meta vector instead of words, **our method inherently addresses the linear growth issue of privacy budget encountered in word-level privatization (Mattern et al., 2022a)**. To further prevent privacy leakage through generated outputs, the server should employ sampling-based generation, enabling the output to be protected by the Exponential Mechanism (Utpala et al., 2023a; Mattern et al., 2022c; McSherry and Talwar, 2007). Experimental results demonstrate that our method can effectively protect private information and maintain satisfactory performance and inference efficiency. The contributions are summarized as follows:

- We propose a plug-and-play privacy protection method that removes privacy spans in the input and restores private information via activation steering during inference.
- We propose Attention-aware Weighted Aggregation to construct the meta vector and apply the d_χ-privacy mechanism to the meta vector, inherently addressing the problem of the linear growth of privacy budget.
- We construct three datasets, covering the medical and legal fields, to evaluate our method. Experimental results demonstrate its capabilities of privacy protection. It also maintains acceptable performance and inference efficiency.

2 Related Works

In this section, we introduce the related works on user input protection methods, which are currently divided into two categories: SMPC-based methods and DP-based methods.

SMPC-based methods. Secure Multi-Party Computation (SMPC) uses encryption algorithms to enable secure collaborative computations between the client and server, without revealing the original user inputs to the server.

However, SMPC incurs significant inference time overhead, rendering it impractical for real-

[2] https://en.wikipedia.org/wiki/Informational_self-determination

time LLM applications (Li et al., 2023a; Liang et al., 2024; Hao et al., 2022a; Liu and Liu, 2023; Zheng et al., 2023b; Gupta et al., 2023; Lu et al., 2024).

DP-based methods. d_χ-privacy mechanisms, a variant of DP, protect user inputs by injecting noise. However, this approach can lead to performance degradation.

Additionally, d_χ-privacy becomes less effective as input length increases, due to the linear growth of the privacy budget (Feyisetan et al., 2019; Mattern et al., 2022b; Utpala et al., 2023b; Dwork et al., 2016; Duchi et al., 2013).

Due to the limited space, a detailed introduction of the above works can be found in Appendix B.

3 Threat Model

We consider a threat model involving two parties: a server that holds the LLM weights and a client holds user inputs containing privacy spans. Privacy span is defined as **a contiguous token sequence that contains private information** in user inputs, and should be identified by the user itself according to the principle of "Information Self-Determination Right". The server provides services through an API while maintaining the confidentiality of the LLM weights. Adversaries may intercept privacy spans when users submit their inputs via the API. Relying solely on encryption algorithms is insufficient to prevent privacy leakage, as encrypted inputs will be decrypted on the server and the server itself may be vulnerable. Some advanced adversaries can even hack the server to steal those decrypted user inputs easily (Abrams, 2024; Toulas, 2024). Therefore, our goal is to protect the privacy spans that exist in user inputs from attackers who are capable of stealing user privacy during transmission or even hacking the server to steal user privacy.

4 Methodology

PrivacyRestore operates in two stages, i.e., the preparation stage and the inference stage, as shown in Figure 1:

(1) **Preparation stage**: This stage takes place only on the server. Considering the long-tailed distribution of privacy spans, we predefine a core set of privacy span types that covers the majority of them. Next, we **identify the edited attention heads** required for activation steering during the inference stage. Finally, we **train the restoration vector** for each privacy span type in the predefined core set. After training, all these vectors are released to the clients. The preparation stage is conducted offline, prior to the server beginning to offer its services.

(2) **Inference stage**: This stage involves collaboration between the client and server. According to the principle of "Information Self-Determination Right", the users should identify all privacy spans in their queries by themselves. Then, the client removes all these privacy spans from the queries for privacy protection. For restoration, the client **constructs a meta vector** according to the removed privacy spans and applies d_χ-privacy to the meta vector to prevent privacy leakage. The meta vector, along with the incomplete queries with privacy spans removed, are sent to the server. The server performs inference on the incomplete input and **restores information** using the meta vector through activation steering.

The preparation and inference stages descriptions are provided in §4.1 and §4.2, respectively. All notation definitions are shown in Appendix A. Backgrounds about the d_χ-privacy mechanism and activation steering are shown in Appendix C.

4.1 Preparation Stage

Edited Heads Identification. As indicated by activation steering methods (Li et al., 2023c; Chen et al., 2024), modifying all attention heads in LLMs will degrade overall performance. Inspired by this, we aim to identify the attention heads most relevant to privacy spans.

As shown in the upper part of Figure 1, we firstly utilize the probe technique (Alain and Bengio, 2016; Tenney et al., 2019; Belinkov, 2022) to identify the most relevant attention heads for each privacy span type. We train a binary classifier for each head, tailored to the privacy span type c, as the probe. A probe with higher accuracy indicates a stronger correlation between the head h and the privacy span type c. Therefore, we select the top K attention heads with the highest accuracies for each privacy span type c in the predefined core set $\mathcal{C}$.

Using different top-K head sets for different privacy span types may suffer the risk of privacy leakage, as an attacker could infer the presence of a specific privacy span type based on the characteristics of top-K head set. Hence, we propose a **Common Top-K Selector** to combine all different top-K head sets to construct a common top-K head set $\mathcal{H}_k$ as

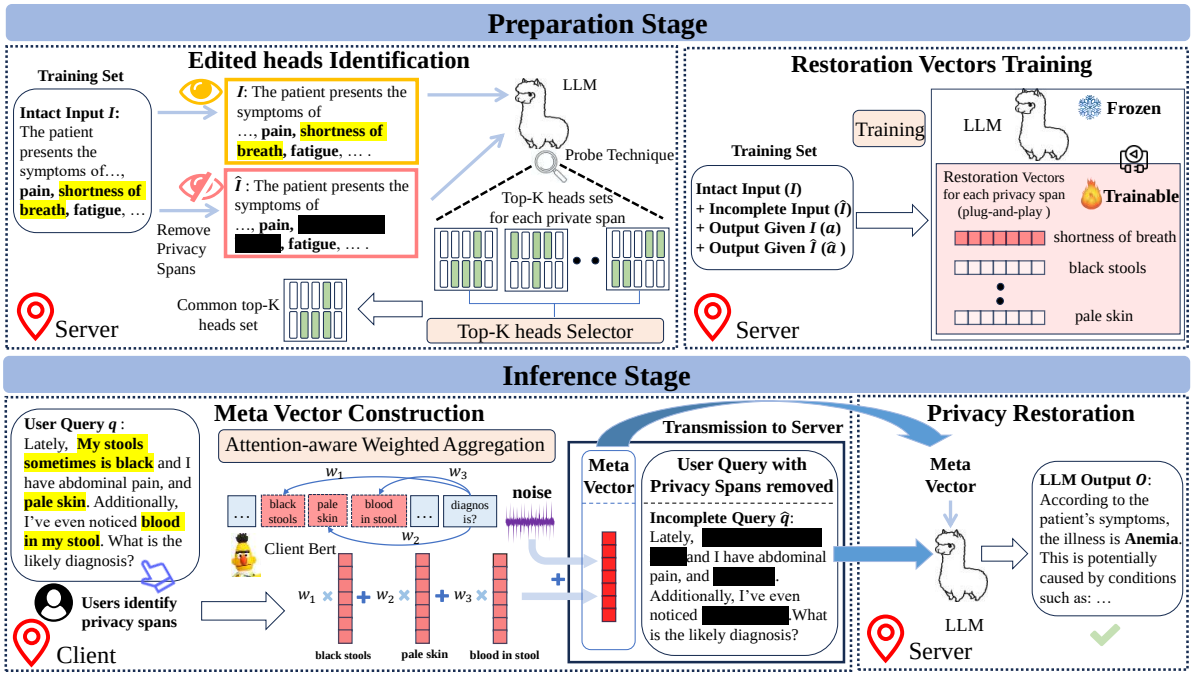


Figure 1: The PrivacyRestore consists of two stages. (1) **Preparation Stage.** This stage aims to identify the edited heads and train the restoration vectors. We provide a more detailed training set example in Figure 4 in the Appendix. (2) **Inference Stage.** In this stage, the client constructs a meta vector. The server uses the meta vector to restore information during inference on the incomplete query.

the edited head set. To achieve this, we calculate the average score of each head across all privacy span types in $\mathcal{C}$, selecting the highest K heads to construct the common set. A head receives a positive score if it appears in the top-K head set of a privacy span type c. The score is related to the accuracy of the probe associated with the head.

Restoration Vectors Training. After identifying the edited heads set $\mathcal{H}_k$, the next step is to train the restoration vectors for each privacy span type in the predefined core set $\mathcal{C}$.

For each privacy span type $c \in \mathcal{C}$, there is a trainable restoration vector r_h^c for each head h in the common top-K heads set $\mathcal{H}_k$. The restoration vectors constitute the only trainable parameters Θ in our method, while the LLM weights held by the server remain fixed. Therefore, our method is plug-and-play and parameter-efficient for training.

We fine-tune the restoration vectors using the ORPO loss proposed by Hong et al. (2024), which integrates the supervised fine-tuning process and the preference alignment process. This loss function can guide the model in generating better answers. In our method, we use ORPO loss to train the restoration vectors Θ, ensuring that the outputs generated from inputs without privacy spans and restored using the corresponding restoration vectors, can closely resemble those generated from the intact inputs.

More details of the probe technique, the common top-K head set construction and restoration vectors training process are provided in Appendix D, Appendix E and Appendix F, respectively. After restoration vectors training, the server will release all restoration vectors to clients.

4.2 Inference Stage

Meta Vector Construction. According to the principle of "Information Self-Determination Right", users should identify the privacy spans in their input by themselves, because the definition of privacy varies from person to person. For each privacy span, the client employs a lightweight model (e.g. BERT (Devlin et al., 2019)) to classify it into a specific type within the predefined privacy span type set $\mathcal{C}$. For example, the privacy span "*My stools sometimes is black*" will be classified into the predefined privacy span type *black stools*. Due to the long-tailed distribution of privacy spans, our predefined type set can cover the majority of privacy spans. Even when encountering privacy spans of out-of-set types, classifying these rare spans into the types of the predefined set can still be effective, as shown in §6.6.

Then the client should aggregate those restoration vectors corresponding to the privacy spans

into a single meta vector. Transmitting a single meta vector enhances privacy protection compared to multiple vectors, as it prevents potential information leakage regarding the quantity of privacy-sensitive segments.

While Equal Weighted Aggregation (EWA) offers a straightforward solution for aggregation, it may weaken the influence of critical privacy spans and amplify the effect of irrelevant ones. Therefore, we propose a novel method called **Attention-aware Weighted Aggregation (AWA)** which estimates a weight for each privacy span and then takes the weighted sum of corresponding restoration vectors as the meta vector. Given computational constraints on client devices, we employ a lightweight BERT model to evaluate privacy span significance by calculating the mean attention score w_s across all attention heads and tokens within the input query. This metric quantifies the relative importance of each privacy-related span.

Considering that each privacy span type c will have multiple restoration vectors r_h^c across all edited heads in $\mathcal{H}_k$, we first concatenate these restoration vectors from multiple heads to form r^c for privacy span type c. Then, we compute the meta vector $\mathcal{R}$ by calculating the weighted sum of the restoration vector r^c, normalizing the summary, and adding noise $\mathcal{N}$ for privacy protection. The process is formulated as follows:

$$r^c = \text{Concat}(r_1^c, r_2^c, ..., r_h^c), \quad (1)$$

$$Z = \frac{\sum_{s \in \mathcal{S}_q} w_s \cdot r^c}{||\sum_{s \in \mathcal{S}_q} w_s \cdot r^c||_2}, \quad (2)$$

$$\mathcal{R} = Z + \mathcal{N}, \quad (3)$$

where s represents the privacy span of type c, $\mathcal{S}_q$ denotes all privacy spans in the user query q and Z represents the normalization of the weighted sum, which can also be viewed as the meta vector without protection. The injected noise $\mathcal{N}$ is sampling from the distribution $p(\mathcal{N}) \propto \exp(-\epsilon\|\mathcal{N}\|)$, to achieve the d_χ-privacy mechanism, where ϵ is the privacy hyperparameter (Feyisetan et al., 2020).

After construction, the meta vector $\mathcal{R}$ and the incomplete query $\hat{q}$ (with privacy spans removed) are transmitted to the server for inference.

Privacy Restoration. We utilize the meta vector $\mathcal{R}$ to restore the information in the removed privacy spans during inference, as illustrated in the lower right part of Figure 1. This operation is conducted on the server side.

Following activation steering methods (Li et al., 2023c; Chen et al., 2024), we apply the meta vector to the outputs of the edited attention heads to achieve restoration. Let $\mathbf{u}_h$ represent the hidden state of the last token on head h given the incomplete user query $\hat{q}$ and $\mathcal{R}_h$ denotes a part of the meta vector $\mathcal{R}$ for head h. Then the hidden state of the last token on head h after restoration, denoted as $\bar{\mathbf{u}}_h$, can be computed by:

$$\bar{\mathbf{u}}_h = \mathbf{u}_h + ||\mathbf{u}_h||_2 \cdot \mathcal{R}_h, \ \forall h \in \mathcal{H}_k. \quad (4)$$

During inference, if a head belongs to the common top-K heads set $\mathcal{H}_k$, its hidden state should be modified using Eq 4. To prevent privacy leakage from the generated output, we employ sampling-based generation, which is protected by the Exponential Mechanism (Utpala et al., 2023a).

5 Privacy Guarantee Analysis

In this section, we analyze the privacy guarantees and privacy budget of PrivacyRestore.

Our approach transmits only a privacy-free incomplete query and a meta vector secured by d_χ-privacy mechanism. Therefore, even if attackers steal both the incomplete user query and the meta vector during transmission or even hack the server, they still cannot infer any user privacy. Furthermore, the confidentiality of the server's LLM parameters and edited head set $\mathcal{H}_k$ prevents attackers from reconstructing the generation process using intercepted elements, ensuring robust security against privacy breaches. Then we analyze the privacy budget of our method, as follows:

Theorem 5.1. *PrivacyRestore fulfills d_χ-privacy and provides a privacy budget of 2ϵ, where ϵ denotes privacy hyperparameter. The privacy budget of PrivacyRestore is independent of the length of the protected text.*

Pointed by Mattern et al. (2022b), directly applying d_χ-privacy mechanism to all tokens in the user query, for privacy protection, suffers from the linear growth problem of privacy budget. In contrast, our method ensures that the privacy budget remains constant at 2ϵ, independent of the length of protected text. We also provide empirical evidence to demonstrate that our approach effectively addresses the linear growth problem of the privacy budget encountered in d_χ-privacy in §6.4. Detailed proof of Theorem 5.1 is provided in Appendix H.

6 Experiments

6.1 Experiments Setup

Datasets. To evaluate our method, we construct three privacy-preserving datasets covering the medical and legal domains: **Pri-DDXPlus**, **Pri-NLICE**, and **Pri-SLJA**. The detailed process of dataset construction and statistical information can be found in Appendix J.

Metrics. The evaluation assesses both performance and inference efficiency. For performance evaluation, we use **MC1/MC2** (Zhang et al., 2024), ROUGE-L (Lin, 2004), and **LLM-Judge (LLM-J)** (Zheng et al., 2023a) metrics. For inference efficiency, we use the **Throughput (TP)** metric. The details of these metrics and their corresponding calculation processes are provided in Appendix L.1.

Compared Methods. To demonstrate the effectiveness of our method, we compare our model with the following baselines: **No Protection**, **No Restoration**, **d_χ-privacy** (Feyisetan et al., 2020), **d_χ-privacy on privacy spans** and **Paraphrase** (Mattern et al., 2022b; Utpala et al., 2023b). A detailed introduction to these baseline methods is provided in Appendix L.2.

Settings of Privacy Hyperparameters. The hyperparameters related to privacy protection strength are ϵ for d_χ-privacy (on privacy spans) and PrivacyRestore, and τ for paraphrase. For a fair comparison, we ensure all methods are under the same privacy budget. We show the calculation process of determining values of ϵ and τ for different methods on different datasets in Appendix M.

6.2 Main Results

As shown in Table 1, we evaluate the performance and inference efficiency of PrivacyRestore and other compared methods across three privacy-preserving datasets. Compared to d_χ-privacy and paraphrase, d_χ-privacy on privacy spans solely apply d_χ-privacy mechanism to those privacy spans and achieves higher scores in MC1/2, ROUGE-L and LLM-J. The possible reason for this is that both d_χ-privacy and paraphrase operate on the entire user input, instead of specific privacy spans. Injecting noise into the entire input creates larger disturbances during inference compared to only corrupting a limited number of privacy spans.

PrivacyRestore achieves best scores in MC1/2 and LLM-J compared to other privacy-preserving methods. In terms of the ROUGE-L evaluation metric, PrivacyRestore achieve the best result in Pri-NLICE while ranking second in the other two datasets. This discrepancy likely stems from ROUGE-L's dependence on n-gram overlap between the reference text and the generated output, which does not fully reflect the quality of generated outputs. As demonstrated by the examples in Figure 8 and Appendix W, PrivacyRestore often generates outputs with different sentence structures while still providing accurate answers. Consequently, our method achieves slightly lower ROUGE-L scores but significantly higher LLM-J scores compared to d_χ-privacy on privacy spans. Furthermore, the ROUGE-L metric displays larger variance than the LLM-J metric, potentially due to its sensitivity to expression rather than the underlying meaning of the generated output. Shown in Table 1, No Protection servers as the performance **upper bound** for all privacy-preserving methods while No Restoration servers as the performance **lower bound**. Our method significantly outperforms No Restoration and is even comparable to No Protection, strongly validating the effectiveness of our approach.

Although PrivacyRestore incurs slight latency from client-side privacy span identification and meta-vector construction, its throughtput attain nearly 70% of the best results, which is acceptable. We provide further analysis and additional experimental results in Appendix V.

6.3 Empirical Privacy Protection Results

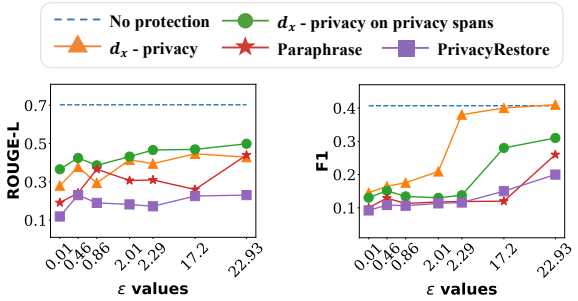
In this section, we implement attack methods to empirically show that our approach offers superior privacy protection compared to baselines, both for user inputs and model outputs.

Privacy Protection Evaluation on Inputs. In this section, we implement the embedding inverse attack (Li et al., 2023b; Morris et al., 2023) and attribute inference attack (Li et al., 2022) to attack the inputs of our method and other baselines, including the meta vector and the privacy-free incomplete user query. As shown in Figure 2, as the privacy budget increases, the privacy protection capability of all privacy-preserving methods decreases. However, PrivacyRestore consistently outperforms others across all privacy budgets, as indicated by its lower ROUGE-L and F1 scores.

Privacy Protection Evaluation on Outputs. We use the sampling-based method to generate the outputs on the server. As demonstrated by Utpala et al. (2023a); Mattern et al. (2022b), sampling-

Datasets	Methods	MC1 ↑	MC2 ↑	ROUGE-L ↑	LLM-J ↑	TP ↑
Pri-DDXPlus	No Restoration (lower bound)	$33.57_{\pm 0.00}$	$32.49_{\pm 0.01}$	$25.19_{\pm 0.43}$	$3.21_{\pm 0.01}$	$40.86_{\pm 0.01}$
	No Protection (upper bound)	$64.88_{\pm 0.01}$	$61.48_{\pm 0.03}$	$100.00_{\pm 0.00}$	$5.58_{\pm 0.03}$	**41.08**$_{\pm 0.09}$
	d_χ-privacy	$28.79_{\pm 0.02}$	$30.26_{\pm 0.01}$	$17.97_{\pm 0.00}$	$1.17_{\pm 0.00}$	$37.45_{\pm 0.01}$
	d_χ-privacy on privacy spans	$44.71_{\pm 0.29}$	$42.36_{\pm 0.00}$	**29.17**$_{\pm 0.04}$	$3.31_{\pm 0.00}$	$33.21_{\pm 0.00}$
	Paraphrase	$27.92_{\pm 0.56}$	$28.56_{\pm 0.07}$	$18.04_{\pm 0.01}$	$1.23_{\pm 0.00}$	$35.42_{\pm 0.67}$
	PrivacyRestore	**62.97**$_{\pm 0.00}$	**60.19**$_{\pm 0.00}$	$27.24_{\pm 0.26}$	**4.47**$_{\pm 0.00}$	$26.09_{\pm 0.08}$
Pri-NLICE	No Restoration (lower bound)	$27.07_{\pm 1.98}$	$28.63_{\pm 2.23}$	$16.90_{\pm 0.51}$	$1.61_{\pm 0.03}$	$41.08_{\pm 0.01}$
	No Protection (upper bound)	$80.30_{\pm 0.38}$	$77.60_{\pm 1.23}$	$100.00_{\pm 0.00}$	$5.90_{\pm 0.04}$	**41.44**$_{\pm 0.04}$
	d_χ-privacy	$29.08_{\pm 0.00}$	$29.72_{\pm 0.00}$	$15.68_{\pm 0.02}$	$1.41_{\pm 0.00}$	$38.30_{\pm 0.00}$
	d_χ-privacy on privacy spans	$30.00_{\pm 0.09}$	$31.46_{\pm 0.00}$	$22.97_{\pm 0.00}$	$3.01_{\pm 0.00}$	$35.73_{\pm 0.57}$
	Paraphrase	$28.46_{\pm 0.02}$	$29.15_{\pm 0.03}$	$16.15_{\pm 0.01}$	$1.62_{\pm 0.00}$	$37.22_{\pm 0.07}$
	PrivacyRestore	**62.23**$_{\pm 1.70}$	**57.94**$_{\pm 0.09}$	**24.42**$_{\pm 0.81}$	**3.67**$_{\pm 0.01}$	$32.33_{\pm 0.01}$
Pri-SLJA	No Restoration (lower bound)	$24.92_{\pm 0.98}$	$25.97_{\pm 1.12}$	$31.02_{\pm 0.16}$	$4.43_{\pm 0.01}$	$39.14_{\pm 0.09}$
	No Protection (upper bound)	$69.57_{\pm 0.61}$	$67.58_{\pm 0.43}$	$100.00_{\pm 0.00}$	$5.44_{\pm 0.03}$	**39.49**$_{\pm 0.13}$
	d_χ-privacy	$16.66_{\pm 0.37}$	$17.57_{\pm 0.04}$	$23.35_{\pm 0.00}$	$2.08_{\pm 0.00}$	$36.83_{\pm 0.03}$
	d_χ-privacy on privacy spans	$24.23_{\pm 1.69}$	$26.63_{\pm 0.67}$	**40.10**$_{\pm 0.00}$	$4.54_{\pm 0.00}$	$36.16_{\pm 0.00}$
	Paraphrase	$16.21_{\pm 0.02}$	$17.52_{\pm 0.02}$	$24.90_{\pm 0.01}$	$2.07_{\pm 0.01}$	$31.31_{\pm 0.05}$
	PrivacyRestore	**35.47**$_{\pm 1.48}$	**35.41**$_{\pm 0.64}$	$37.56_{\pm 0.06}$	**5.25**$_{\pm 0.00}$	$30.73_{\pm 0.04}$

Table 1: Comparison of the performance and the inference efficiency between PrivacyRestore and other baselines across three privacy-preserving datasets. All experiments are conducted over 3 runs, with the average results and variances reported. The best results are highlighted in **bold**.



(a) Embedding Inverse Attack (b) Attribute Inference Attack

Figure 2: Results of embedding inverse attack and attribute inference attack for all baselines under different privacy hyperparameters ϵ on Pri-DDXPlus.

based generation satisfies the Exponential Mechanism (McSherry and Talwar, 2007), which can effectively prevent privacy leakage from the generated outputs. We implement the embedding inversion and attribute inference attacks for the generated outputs under various generation temperatures and also count the occurrence of privacy spans in the outputs. As shown in Table 2, the attack performance remains consistently low, demonstrating that sampling-based generation effectively prevents privacy leakage from the generated outputs. Implementation details of these attack methods can be found in Appendix O and Appendix Q.

We also provide a biref theoretical proof of our method's output protection in Appendix I.

Temperature	0.75	1.0	1.25	1.5	1.75
EIA(ROUGE-L)	0.037	0.038	0.035	0.035	0.037
AIA(F1)	0.096	0.097	0.092	0.092	0.097
Occurrence	0.031	0.030	0.030	0.029	0.031

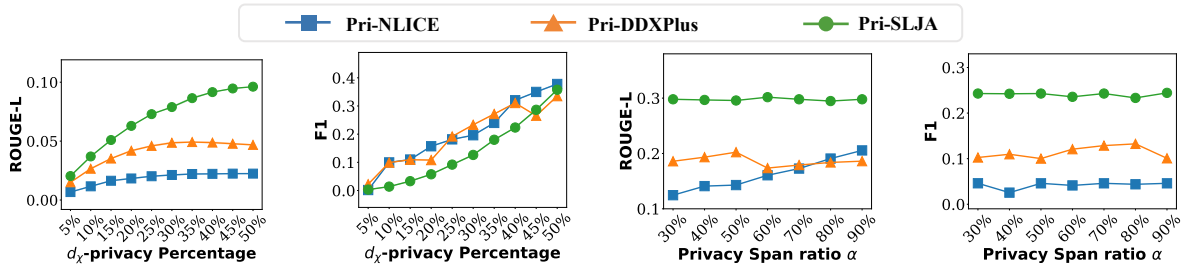
Table 2: Analysis of output privacy leakage from outputs on Pri-DDXPlus dataset. **EIA** denotes embedding inverse attack. **AIA** indicates attribute inference attack. **Occurrence** metric directly counts the frequency of privacy spans in the generated output. We primarily use a temperature of 1.0 during generation in the other experiments.

6.4 Privacy Protection for Long Text

In this section, we implement attack methods for both the d_χ-privacy baseline and our PrivacyRestore approach under varying protected text lengths, illustrating robust privacy and addressing the linear growth of the privacy budget in d_χ-privacy.

For **d_χ-privacy**, we randomly select a proportion of tokens to protect. Note that higher percentages yield longer protected text. As shown in Figures 3(a) and 3(b), both prompt injection and attribute inference attacks demonstrate better attack performance with longer protected text. It is caused by the linear growth problem of the privacy budget encountered in d_χ-privacy, as raised by Mattern et al. (2022b).

For **PrivacyRestore**, a proportion of privacy spans is selected for protection, defined by the Privacy Span Ratio α, with larger α indicating more



(a) Prompt Injection Attack (b) Attribute Inference Attack (c) Embedding Inverse Attack (d) Attribute Inference Attack

Figure 3: (a) and (b) present the results of d_χ-privacy method under the prompt injection attack and attribute inference attack under varying d_χ-privacy percentages across three privacy-preserving datasets. (c) and (d) show the results of PrivacyRestore for the embedding inverse attack and attribute inference attack under different privacy span ratios α on the same three datasets.

spans. Shown in Figures 3(c) and 3(d), aside from the embedding inverse attack on the Pri-NLICE dataset, attack performance remains stable across different α values. These results confirm that our method provides strong privacy protection, even as text length increases.

6.5 Ablation study

Attention-Aware Weighted Aggregation. To assess the effectiveness of the Attention-Aware Weighted Aggregation (AWA) component, we compare its performance and inference efficiency against Equal Weighted Aggregation (EWA). Unlike AWA, EWA generates the meta vector by summing all restoration vectors equally. As shown in Table 3, EWA results in lower MC1, MC2, ROUGE-L, and LLM-J scores compared to AWA, indicating that equal weighting will diminish performance by amplifying irrelevant privacy spans.

Datasets	Methods	MC1 ↑	MC2 ↑	ROUGE-L ↑	LLM-J ↑	TP ↑
Pri-DDXPlus	EWA	53.84	51.12	26.32	4.29	**26.35**
	AWA	**62.97**	**60.19**	**27.24**	**4.47**	26.09
Pri-NLICE	EWA	46.92	45.89	22.78	3.12	**32.75**
	AWA	**62.23**	**57.94**	**24.42**	**3.67**	32.33
Pri-SLJA	EWA	30.88	30.70	30.96	4.10	**31.00**
	AWA	**35.47**	**35.41**	**37.56**	**5.25**	30.73

Table 3: Comparison of the performance and the inference efficiency between Equal Weighted Aggregation (**EWA**) and Attention-aware Weighted (**AWA**) Aggregation. The best results are highlighted in **bold**.

Other Ablation Studies. Furthermore, we evaluate PrivacyRestore's performance with varying numbers of edited heads (K) and with an alternative LLM backbone (Llama-13b-chat) in Appendix U. The results in Table 15 and Figure 7 clearly demonstrate the effectiveness of our method.

6.6 Extension Analysis of PrivacyRestore

In this section, we analyze the extension PrivacyRestore to other more extreme scenarios.

Encountering Out-of-Set Privacy Spans. Due to the long-tailed distribution of privacy spans shown in Appendix K, the core set covers most spans. We further evaluated our method when encountering out-of-set spans. Specifically, we include only a subset of privacy span types in our core set. Table 11 shows that our method still demonstrates superior performance, compared to No Restoration baseline. More implementation details are shown in Appendix S.

Users Unable to Determine Privacy Spans. Our method follows the principle of "Information Self-Determination Rights" allowing users to determine their own privacy spans. Even when users cannot or choose not to specify these spans, our method remains effective by integrating with existing **text sanitization techniques** (Kan et al., 2023; Chen et al., 2023a). As shown in Table 14, our method can maintains superior performance, and details of implementation are provided in Appendix T.

7 Conclusion

We propose PrivacyRestore which protects the privacy within user inputs during inference in online LLM inference services. PrivacyRestore achieves privacy protection by directly removing privacy spans in the user input and then restoring information via activation steering. PrivacyRestore provides a practical and efficient solution for protecting privacy while maintaining satisfactory performance and inference efficiency. We demonstrate that PrivacyRestore inherently addresses the linear growth problem of the privacy budget found

in d_χ-privacy. We curate three privacy-preserving datasets covering medical and legal fields, and PrivacyRestore achieves strong performance and inference efficiency across all datasets. Additionally, we implemented various attack methods, and the attack results demonstrate PrivacyRestore's robust privacy protection capabilities.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (62406114, 62306117, 62472181), the National KeyR&D Project fromMinister of Science and Technology (2024YFA1211500), the Fundamental Research Funds for the Central Universities (2024ZYGXZR074), the Guangdong Basic and Applied Basic Research Foudation (2025A1515011413, 2024A1515010220), the Guangzhou Basic and Applied Basic Research Foundation (2024A04J3681), the GJYC program of Guangzhou (2024D03J0005) and the South China University of Technology-TCL Technology Innovation Fund.

The work described in this paper was conducted in full or in part by Dr. Haoran Li, JC STEM Early Career Research Fellow, supported by The Hong Kong Jockey Club Charities Trust.

Limitations

This section aims to highlight the limitations of our work and provide further insights into the research in this area.

One limitation is that we only evaluate our method in the medical and legal domains and additional domains could be explored to validate its effectiveness.

Another limitation is that more attack methods could be explored to assess the privacy protection of our approach. While we have implemented most of the current advanced attack methods, to the best of our knowledge, there may be others yet to be tested. Additionally, more advanced attack methodologies may emerge in the future, which will also need to be evaluated.

Ethics Statement

We adhere to the ACL Ethics Policy and have conducted our research using publicly available repositories and datasets. In the PrivacyRestore framework, we have adhered to rigorous ethical standards to safeguard user privacy and uphold data security. All three datasets (Pri-SLJA, Pri-NLICE and Pri-DDXPlus) utilized in this research are sourced exclusively from publicly available repositories, ensuring that these datasets are devoid of any personally identifiable information and minimizing potential privacy risks. Our methodology does not access or reconstruct the original identifiable data or its sources. This ensures that the research does not infringe upon individual privacy rights.

However, due to the fact that we employed multiple LLMs in this study, such as ChatGPT, Qwen and GPT-4. The findings may be influenced by the inherent assertiveness, linguistic patterns, and diverse biases characteristic of these LLMs.

References

Lawrence Abrams. 2024. UnitedHealth says data of 100 million stolen in Change Healthcare breach. from bleepingcomputer.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Zaid Al-Ars, Obinna Agba, Zhuoran Guo, Christiaan Boerkamp, Ziyaad Jaber, and Tareq Jaber. 2023. Nlice: Synthetic medical record generation for effective primary healthcare differential diagnosis. In *2023 IEEE 23rd International Conference on Bioinformatics and Bioengineering (BIBE)*, pages 397–402. IEEE.

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

B Van Alsenoy, E Kosta, and J Dumortier. 2014. Privacy notices versus informational self-determination: Minding the gap. International Review of Law, Computers & Technology.

Mário S. Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. 2018. Invited paper: Local differential privacy on metric spaces: Optimizing the trade-off with utility. In *31st IEEE Computer Security Foundations Symposium, CSF 2018, Oxford, United Kingdom, July 9-12, 2018*, pages 262–267.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

Haohan Bo, Steven H. H. Ding, Benjamin C. M. Fung, and Farkhund Iqbal. 2021. ER-AE: differentially private text generation for authorship anonymization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3997–4007.

Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 2633–2650. USENIX Association.

Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the scope of differential privacy using metrics. In *Privacy Enhancing Technologies - 13th International Symposium, PETS 2013, Bloomington, IN, USA, July 10-12, 2013. Proceedings*, pages 82–102.

Yu Chen, Tingxin Li, Huiming Liu, and Yang Yu. 2023a. Hide and seek (has): A lightweight framework for prompt privacy protection. *ArXiv*, abs/2309.03057.

Zeming Chen, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, et al. 2023b. Meditron-70b: Scaling medical pretraining for large language models. *arXiv preprint arXiv:2311.16079*.

Zhongzhi Chen, Xingwu Sun, Xianfeng Jiao, Fengzong Lian, Zhanhui Kang, Di Wang, and Chengzhong Xu. 2024. Truth forest: Toward multi-scale truthfulness in large language models through intervention without tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 20967–20974.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2024. Scaling instruction-finetuned language models. *J. Mach. Learn. Res.*, 25:70:1–70:53.

Wentao Deng, Jiahuan Pei, Keyi Kong, Zhe Chen, Furu Wei, Yujun Li, Zhaochun Ren, Zhumin Chen, and Pengjie Ren. 2023. Syllogistic reasoning for legal judgment analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 13997–14009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2013. Local privacy and statistical minimax rates. In *51st Annual Allerton Conference on Communication, Control, and Computing, Allerton 2013, Allerton Park & Retreat Center, Monticello, IL, USA, October 2-4, 2013*, page 1592.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2016. Calibrating noise to sensitivity in private data analysis. *J. Priv. Confidentiality*, 7(3):17–51.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations.

In *Proceedings of the 13th international conference on web search and data mining*, pages 178–186.

Oluwaseyi Feyisetan, Tom Diethe, and Thomas Drake. 2019. Leveraging hierarchical representations for preserving privacy and utility in text. In *2019 IEEE International Conference on Data Mining, ICDM 2019, Beijing, China, November 8-11, 2019*, pages 210–219.

Ahmed Frikha, Nassim Walha, Krishna Kanth Nakka, Ricardo Mendes, Xue Jiang, and Xuebing Zhou. 2024. Incognitext: Privacy-enhancing conditional text anonymization via llm-based private attribute randomization. *CoRR*, abs/2407.02956.

Kanav Gupta, Neha Jawalkar, Ananta Mukherjee, Nishanth Chandran, Divya Gupta, Ashish Panwar, and Rahul Sharma. 2023. Sigma: Secure gpt inference with function secret sharing. Cryptology ePrint Archive, Paper 2023/1269. https://eprint.iacr.org/2023/1269.

Meng Hao, Hongwei Li, Hanxiao Chen, Pengzhi Xing, Guowen Xu, and Tianwei Zhang. 2022a. Iron: Private inference on transformers. In *Advances in Neural Information Processing Systems*, volume 35, pages 15718–15731. Curran Associates, Inc.

Meng Hao, Hongwei Li, Hanxiao Chen, Pengzhi Xing, Guowen Xu, and Tianwei Zhang. 2022b. Iron: Private inference on transformers. *Advances in neural information processing systems*, 35:15718–15731.

Evan Hernandez, Belinda Z. Li, and Jacob Andreas. 2023. Inspecting and editing knowledge representations in language models. *Preprint*, arXiv:2304.00740.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. ORPO: monolithic preference optimization without reference model. *CoRR*, abs/2403.07691.

Jasper M C. 2009. Privacy and the Internet Your Expectations and Rights Under the Law. Oxford University Press, Inc.

Zhigang Kan, Linbo Qiao, Hao Yu, Liwen Peng, Yifu Gao, and Dongsheng Li. 2023. Protecting user privacy in remote conversational systems: A privacy-preserving framework based on text sanitization. *ArXiv*, abs/2306.08223.

Dacheng Li, Hongyi Wang, Rulin Shao, Han Guo, Eric Xing, and Hao Zhang. 2023a. MPCFORMER: FAST, PERFORMANT AND PRIVATE TRANSFORMER INFERENCE WITH MPC. In *The Eleventh International Conference on Learning Representations*.

Haoran Li, Yangqiu Song, and Lixin Fan. 2022. You don't know my favorite color: Preventing dialogue representations from revealing speakers' private personas. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 5858–5870.

Haoran Li, Mingshi Xu, and Yangqiu Song. 2023b. Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 14022–14040.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023c. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, pages 41451–41530. Curran Associates, Inc.

Zi Liang, Pinghui Wang, Ruofei Zhang, Nuo Xu, Shuo Zhang, Lifeng Xing, Haitao Bai, and Ziyang Zhou. 2024. Merge: Fast private text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19884–19892.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Xuanqi Liu and Zhuotao Liu. 2023. Llms can understand encrypted prompt: Towards privacy-computing friendly transformers. *arXiv preprint arXiv:2305.18396*.

Zhengdong Lu, Ziqian Zeng, Jianwei Wang, Hanlin Wang, Weikai Lu, and Huiping Zhuang. 2024. Zero-shot event argument extraction by disentangling trigger from argument and role. *International Journal of Machine Learning and Cybernetics*, pages 1–19.

Justus Mattern, Zhijing Jin, Benjamin Weggenmann, Bernhard Schoelkopf, and Mrinmaya Sachan. 2022a. Differentially private language models for secure data sharing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. 2022b. The limits of word level differential privacy. In *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 867–881.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. 2022c. The limits of word level differential privacy. In *NAACL-HLT*.

Frank McSherry and Kunal Talwar. 2007. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103.

John X. Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M. Rush. 2023. Text embeddings reveal (almost) as much as text. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 12448–12460.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

Robin Staab, Mark Vero, Mislav Balunovic, and Martin T. Vechev. 2024. Large language models are advanced anonymizers. *CoRR*, abs/2402.13846.

Xuchen Suo. 2024. Signed-prompt: A new approach to prevent prompt injection attacks against llm-integrated applications. *CoRR*, abs/2401.07612.

Arsène Fansi Tchango, Rishab Goel, Zhi Wen, Julien Martel, and Joumana Ghosn. 2022. Ddxplus: A new dataset for automatic medical diagnosis. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Bill Toulas. 2024. Anna Jaques Hospital ransomware breach exposed data of 300K patients. from bleepingcomputer.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. 2023. Activation addition: Steering language models without optimization. *Preprint*, arXiv:2308.10248.

Saiteja Utpala, Sara Hooker, and Pin Yu Chen. 2023a. Locally differentially private document generation using zero shot prompting. In *Conference on Empirical Methods in Natural Language Processing*.

Saiteja Utpala, Sara Hooker, and Pin-Yu Chen. 2023b. Locally differentially private document generation using zero shot prompting. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 8442–8457.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Shijie Wu, Ozan Irsoy, Steven Lu, Vadim Dabravolski, Mark Dredze, Sebastian Gehrmann, Prabhanjan Kambadur, David Rosenberg, and Gideon Mann. 2023. Bloomberggpt: A large language model for finance. *arXiv preprint arXiv:2303.17564*.

Yiquan Wu, Yuhang Liu, Yifei Liu, Ang Li, Siying Zhou, and Kun Kuang. wisdominterrogatory. Available at GitHub.

Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. 2020. A differentially private text perturbation method using a regularized mahalanobis metric. *CoRR*, abs/2010.11947.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Shaolei Zhang, Tian Yu, and Yang Feng. 2024. Truthx: Alleviating hallucinations by editing large language models in truthful space. *arXiv preprint arXiv:2402.17811*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023a. Judging llm-as-a-judge with mt-bench and chatbot arena. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Mengxin Zheng, Qian Lou, and Lei Jiang. 2023b. Primer: Fast private transformer inference on encrypted data. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE.

Appendix Overview

The appendix is divided into two parts: Appendix A–I provide backup theoretical explanations of PrivacyRestore, while Appendix J–X present additional experimental results on PrivacyRestore from different aspects.

A Notations

Here we present all notations used in our paper in Table 4.

B Related Works

In this section, we introduce the related works on user input protection methods, which are currently divided into two categories: SMPC-based methods and DP-based methods.

Here, we provide a more detailed introduction to Secure Multi-Party Computation (SMPC) and Differential Privacy (DP).

B.1 Secure Multi-Party Computation (SMPC)

Secure multi-party computation (SMPC) methods utilize multi-party encryption algorithms to enable collaborative computation among multiple parties while protecting the privacy of their data.

However, most nonlinear operations in LLMs cannot directly support secure multi-party computation.

To address this challenge, current SMPC methods focus on two optimization directions: model structure-oriented optimization and protocol-oriented optimization.

The model structure-oriented approach aims to replace SMPC-unfriendly nonlinear operations with SMPC-friendly alternatives.

For instance, MPC-Former (Li et al., 2023a) approximates nonlinear operations in Transformer using polynomials and maintains performance through model distillation.

MERGER (Liang et al., 2024) integrates previous techniques to natural language generation (NLG) tasks by bypassing embedded computation and reorganizing linear operations in Transformer modules, further enhancing computational efficiency and model performance.

In contrast, the protocol-oriented approach focuses on designing efficient SMPC operators for nonlinear operations in LLMs while preserving the original model structure.

Recent works (Hao et al., 2022a; Liu and Liu, 2023; Zheng et al., 2023b; Gupta et al., 2023) have improved the efficiency of nonlinear operations in privacy-preserving LLMs inference by utilizing various SMPC protocols, such as confusion circuit and function secret sharing.

Although SMPC-based methods can be applied to protect user inputs during model inference, they still suffer from large inference time overhead.

For example, inference on the RoBERTa-Base model takes 168.43 seconds (Hao et al., 2022b), making current SMPC methods impractical for online LLM inference services.

B.2 Differential Privacy (DP)

Differential Privacy (DP), as introduced by Dwork et al. (2016), is designed to protect individual privacy by preventing attackers from identifying specific participants in a dataset.

Several variants of DP have been developed to enhance privacy protection across various settings, adapting the core principles of DP to different types of data and threat models.

Notable examples include Centralized Differential Privacy (CDP), Local Differential Privacy (LDP), and d_χ-privacy.

CDP (Dwork et al., 2016) operates under the assumption that all data has been stored in a central repository.

It guarantees that attackers cannot distinguish between any two adjacent repositories based on query results.

In contrast, LDP (Duchi et al., 2013) provides a stronger guarantee, ensuring that attackers cannot distinguish between any two adjacent inputs. Mattern et al. (2022b) and Utpala et al. (2023b) propose using paraphrasing techniques to achieve LDP on user inputs.

d_χ-privacy (Feyisetan et al., 2019), a relaxed version of LDP, incorporates metrics that measure the similarity between inputs, allowing for more flexible control over the privacy budget. The formal definition of d_χ-privacy Mechanism is provided in Appendix C.1.

As proposed by Mattern et al. (2022b), applying d_χ-privacy to all tokens in user inputs, known as word-level privatization, suffers from the linear growth problem of the privacy budget.

This means that as the length of the protected text increases, the privacy protection performance of d_χ-privacy decreases.

Notations	Definitions
c	A privacy span type.
$\mathcal{C}$	All possible privacy span types.
s	A single privacy span.
$\mathcal{S}_q$	All privacy spans in user query q.
h	A single edited head.
$\mathcal{H}_k$	The common top-K heads set.
$\mathcal{H}_a$	The set of all heads.
$\mathcal{H}_k^c$	The top-K heads set of the privacy span type c.
L_h	The score list of the head h across all privacy spans.
K	The number of selected edited heads.
$\mathcal{F}_h^c$	The probe of privacy span type c on head h.
θ_h^c	The parameters of the probe $\mathcal{F}_h^c$.
$\mathbf{u}_h$	The output hidden state on head h.
$\bar{\mathbf{u}}_h$	The output hidden state after restoration on head h.
r_h^c	The restoration vector for privacy span type c on head h.
Θ	All restoration vectors for all privacy spans on all edited heads.
λ	The tradeoff hyperparameter of ORPO loss.
w_s	The weight of privacy span s.
n	The number of tokens in the user query.
n_h	The number of heads in the lightweight model.
$\text{Attn}_h(x, y)$	The attention score of y attending to x on head h.
Z_h, Z_h'	Any two normalized weighted sums of restoration vectors on head h.
$\mathcal{R}$	The meta vector.
$\mathcal{R}_h$	The part of the meta vector for head h.
$\mathcal{N}$	The added noise on the normalized weighted sums for meta vector construction.
I	The user inputs in the training set.
$I_{all} = \{I_1, ..., I_m\}$	All user inputs in the training set.
$Y_c = \{y_1, ..., y_m\}$	The labels indicating whether the corresponding input contains the privacy span of type c.
m	The size of training set.
I, I'	Any two user inputs.
$\{i_1, ..., i_n\}$	The tokens of the input I.
$\{e_1, ..., e_n\}$	Corresponding token embeddings of the input I.
$O = \{o_1, ..., o_n\}$	The possible output sets for I, with each one representing a single output.
$\hat{I}$	The incomplete user input with all privacy spans removed in the training set.
$\hat{I}_{all} = \{\hat{I}_1, ..., \hat{I}_m\}$	All user inputs with privacy spans removed in the training set.
a	The output given the complete input I.
$\hat{a}$	The output given the incomplete input $\hat{I}$.
q	The user query during inference.
$\hat{q}$	The incomplete user query with all privacy spans removed during inference.
ϵ	The privacy hyperparameter.
τ	The generation temperature.
δ	The privacy hyperparameter.
n_{ps}	The number of tokens associated with the privacy spans in the user query.
α	The proportion of privacy spans selected for protection.
d_χ	Any distance function used by d_χ-privacy.
d_e	The distance between token embeddings.
d_z	The distance between normalized weighted sums.

Table 4: Definitions of all notations used in our paper.

C Preliminaries for Methodology

The d_χ-privacy mechanism and activation steering technique are two crucial components of our method. Here, we provide a more detailed illustration of these techniques for a better understanding of our method.

C.1 d_χ-privacy Mechanism

d_χ-privacy mechanism (Feyisetan et al., 2019) is a variant of the differential privacy mechanism designed to protect the privacy and incorporate a distance measure into the privacy budget. The detailed definition of d_χ-privacy is as follows:

Definition C.1. ***(d_χ-privacy mechanism).*** *A randomized mechanism $\mathcal{M} : \mathcal{I} \rightarrow \mathcal{O}$ fulfills ϵ-d_χ-privacy if for all adjacent inputs $I, I' \in \mathcal{I}$ and all possible outputs $O \subset \mathcal{O}$, the following condition holds:*

$$\mathbb{P}\left(\mathcal{M}(I) \in O\right) \leq \exp(\epsilon d_\chi(I, I'))\mathbb{P}\left(\mathcal{M}(I') \in O\right),$$

where d_χ is a distance function defined on $\mathcal{I}$.

Numerous prior works have applied the d_χ-privacy mechanism (Chatzikokolakis et al., 2013; Alvim et al., 2018) to word embeddings to achieve word-level privatization (Feyisetan et al., 2020, 2019; Xu et al., 2020; Bo et al., 2021). In our approach, we employ the d_χ-privacy mechanism to protect the meta vector, preventing privacy leakage from the meta vector.

To implement the d_χ-privacy mechanism on the meta vector/token embeddings, noise must typically be added to it, as shown below:

$$\mathcal{R} = Z + \mathcal{N}, \tag{5}$$

$$\mathbb{P}(\mathcal{N}) \propto \exp(-\epsilon||\mathcal{N}||), \tag{6}$$

where Z is the unprotected meta vector/token embeddings, $\mathcal{N}$ is the added noise, $\mathcal{R}$ is the protected meta vector/token embeddings and ϵ is the privacy parameter of the mechanism. According to Feyisetan et al. (2019), to sample the noise $\mathcal{N}$ from its distribution, we can compute it as follows:

$$\mathbf{v} \in \{v \in \mathbb{R}^n : ||v|| = 1\} \tag{7}$$

$$\mathbb{P}(\mathbf{l}) \propto \frac{\mathbf{l}^{n-1} e^{-\epsilon \mathbf{l}}}{\Gamma(n)\epsilon^{-n}}, \tag{8}$$

$$\mathcal{N} = \mathbf{l} \cdot \mathbf{v}, \tag{9}$$

where n is the size of the meta vector and ϵ is the privacy parameter.

C.2 Activation Steering Technique

Activation steering methods (Li et al., 2023c; Turner et al., 2023; Hernandez et al., 2023) control the behavior of LLM by modifying their activations during the inference stage. It serves as a crucial part of our methodology to restore information contained within the removed privacy spans during LLM inference. Typically, the attention mechanism (Vaswani et al., 2017) in LLM is responsible for capturing contextual information, and it can be expressed as:

$$q = W_q \cdot \mathbf{i}, \tag{10}$$

$$\mathbf{u} = \text{Softmax}(\frac{q \cdot K^T}{\sqrt{d_k}}) \cdot V, \tag{11}$$

where $\mathbf{i}$ is the input hidden state, $\mathbf{u}$ is the output hidden state, W_q is the query weight matrix, K is the key of the context and V is the value of the context and d_k is the dimension of the key. Activation steering methods add some steering vectors into the output hidden state and, in our methods, we add the meta vector into the output hidden state to restore information, which can be expressed as:

$$\mathbf{u} = \mathbf{u} + \mathcal{R}, \tag{12}$$

where $\mathcal{R}$ is the meta vector.

D Selecting the Most Relevant Heads

In this section, we provide the implementation details of the probe technique (Alain and Bengio, 2016; Tenney et al., 2019; Belinkov, 2022) to identify the most relevant attention heads for each type of privacy span.

Let $I_{all} = I_1, ..., I_m$ represent the user inputs in the training set, where m is the size of the training set. For a given privacy span type c, let $Y_c = y_1, ..., y_m$ represent the corresponding labels, where $y_i = 1$ if and only if the input I_i contains a privacy span of type c.

For each user input I_i, we record the hidden state of the last token on each attention head. We then train a binary classifier for each head, tailored to the privacy span type c, as the probe. The probe takes the hidden state of the last token as input and predicts whether the input contains the privacy span of type c. The probe is formulated as:

$$\mathcal{F}_h^c(\mathbf{u}_h) = \sigma(\theta_h^c \cdot \mathbf{u}_h), \tag{13}$$

where $\mathcal{F}_h^c(\cdot)$ is the probe of privacy span type c on head h, $\mathbf{u}_h$ is the hidden state of the last token on

head h, θ_h^c are the parameters of the probe, and $\sigma(\cdot)$ indicates the sigmoid function.

A probe $\mathcal{F}_h^c(\cdot)$ with higher accuracy indicates a stronger correlation between the head h and the privacy span type c. Therefore, we only select the top K attention heads, with the highest accuracies, as the most relevant heads for the privacy span type c.

E The Algorithm of Common Top-K Selector

In this section, we present the detailed implementation of Common Top-K Selector algorithm, as shown in Algorithm 1.

Firstly, we initialize an empty score list L_h for each head. Secondly, each privacy span type c has its corresponding top-K heads set $\mathcal{H}_k^c$. For each head h in $\mathcal{H}_k^c$, we append Score$(h, \mathcal{H}_k^c)$ into its score list L_h. Score$(h, \mathcal{H}_k^c)$ is defined as the rank of head h among $\mathcal{H}_k^c$ in ascending order based on the accuracy of the probe associated with the head h and privacy span type c. Thirdly, we calculate the average value of each score list L_h as the score of the corresponding head h. Finally, we sort all heads in the LLM by the scores and pick up top-K heads as the common top-K head set $\mathcal{H}_k$.

Algorithm 1 Common Top-K Selector

Input: $\mathcal{S}$ is the set of privacy spans; $\mathcal{H}_a$ is the set of all heads; $\mathcal{H}_k^c$ denotes the set of top-K heads corresponding to the privacy span type c; Score$(h, \mathcal{H}_k^c)$ return the rank of head h among $\mathcal{H}_k^c$ in ascending order based on the accuracy of the probe associated with the head h and privacy span type c. The score of the head with lowest accuracy is 1. The score of the head with highest accuracy is K.

```
Initialize an empty score list L_h = [ ] for each head h in H_a.
for c in C do
    for h in H_k^c do
        Append Score(h, H_k^c) into L_h.
    end for
end for
for h in H_a do
    score_h = average(L_h)
end for
Sort H_a according to score_h and select top K heads to obtain common top-K head set H_k.
```

Output: $\mathcal{H}_k$ is the common top-K head set.

F Details of the Training Process

In our method, we use the ORPO loss (Hong et al., 2024) to train the restoration vectors, which are employed to restore information in the removed privacy spans. The training objective is to ensure that, despite receiving incomplete inputs with all privacy spans removed, the model can still generate high-quality outputs similar to those produced from intact inputs by utilizing these restoration vectors.

Assuming that Θ is the trainable restoration vectors, $\hat{I}$ denotes the input with privacy spans removed, $\hat{I}_{all} = \{\hat{I}_1, \cdots, \hat{I}_m\}$ represents the training set of incomplete inputs, a is the initial output give the complete input, $\hat{a}$ is the output given the incomplete input with privacy spans removed, then the training loss of our method can be express as:

$$\begin{aligned} \text{ratio}(a|\hat{I};\Theta) &= \frac{\mathbb{P}(a|\hat{I};\Theta)}{1-\mathbb{P}(a|\hat{I};\Theta)}, \\ \mathcal{L}_{\text{ORPO}} &= \sum_{\hat{I}\in\hat{I}_{all}} -\log \mathbb{P}(a|\hat{I};\Theta) \\ &\quad - \lambda \log \sigma\left(\log \frac{\text{ratio}(a|\hat{I};\Theta)}{\text{ratio}(\hat{a}|\hat{I};\Theta)}\right), \end{aligned} \tag{14}$$

where λ is the hyperparameter that controls the weight of the loss term and the $\mathbb{P}(a|\hat{I};\Theta)$ is the probability of the model generating the initial output a given the intact input after being restored by Θ, and $\mathbb{P}(\hat{a}|\hat{I};\Theta)$ is the probability of generating $\hat{a}$. After training the restoration vectors using the above loss, these vectors can effectively restore the information in the missing privacy spans and guide the model to generate outputs similar to those produced from intact inputs, **even though no privacy spans are present in the input indeed**.

We also provide a training example for better understanding of our training process. As shown in the loss function in Equation 14, each training sample will contain the incomplete input without privacy spans $\hat{I}$, the intact input I, the output a given intact input I and the output given the incomplete input $\hat{I}$. An example of a training sample is presented in Figure 4.

G Usage of the Lightweight Model on the Client Side

As shown in Section 4.2, our method utilizes the lightweight model (e.g. Bert-based-uncased (Devlin et al., 2019)) to classify the privacy spans in

Complete input I : I have a history of chronic pancreatitis, diabetes, and obesity. My family has a history of pancreatic cancer. I experienced a persistent cough, frequent diarrhea, nausea, abdominal pain, pale stools, dark urine, and I've noticed that I've been losing weight unintentionally. I'm concerned about these symptoms and wondering if they could be related to my existing medical conditions or if there might be something else going on. What is the likely diagnosis?

Incomplete input without privacy spans $\hat{I}$: I experienced the symptoms of cough, nausea. I'm concerned about these symptoms and wondering if they could be related to my existing medical conditions or if there might be something else going on. What is the likely diagnosis?

The output a given complete input I: Based on the symptoms and medical history provided, the likely diagnosis for the patient is pancreatic cancer.\n\nPancreatic cancer is a common cause of abdominal pain, weight loss, and gastrointestinal symptoms, including cough, diarrhea, nausea, and pale or dark urine. ...

The output $\hat{a}$ give incomplete input $\hat{I}$: Thank you for the question. Based on the symptoms provided, the likely diagnosis for a person presenting with cough and nausea is a respiratory tract infection, such as bronchitis or pneumonia.\n\nBronchitis is an inflammation of the bronchial tubes ....

Figure 4: A training sample in our framework. Text highlighted with a yellow background represents the privacy spans in user inputs. Text highlighted with a green background indicates the correct diagnosis. Text highlighted with a red background denotes the incorrect diagnosis.

the user query and compute the importance score of these privacy spans for conducting the following Attention-aware Aggregation (AWA). The detailed implementations are as follows:

G.1 For Classifying Privacy Span Types

Each privacy span type can be expressed in various forms within the user query, for example,"*fever*" may be represented as"*elevated body temperature*". After the user identifies the privacy spans in the query, we should classify these spans into those predefined types from the set $\mathcal{C}$.

Firstly, we use a lightweight Bert-based-uncased model on the client side to first extract the vector representation of the privacy span. Specifically, we compute the mean of the hidden states from the last layer across all tokens within the privacy span to obtain the vector representation. We construct a multi-layer perceptron (MLP) classifier, consisting of an input layer, two hidden layers, and an output layer. The MLP classifier takes the vector representation of the privacy span as input, and the output label corresponds to the privacy span type. During the training process, we will fix the Bert-based uncased model while only training the MLP classifier.

G.2 For Computing the Importance Score of Privacy Spans

Each privacy span in the user query should have a distinct importance weight and we also utilized the Bert-base-uncased model to assess the importance weights for the privacy spans. To be specific, we compute the average received attention of privacy span s across all attention heads and all tokens in the user query as the importance score w_s. Assume s is the privacy span s, q is the user query, and then the importance weight of the privacy span w_s is calculated as:

$$w_s = \frac{1}{n}\frac{1}{n_h}\sum_{t=1}^{n}\sum_{h=1}^{n_h}\text{Attn}_h(s, q_t), \quad (15)$$

where n is the number of tokens in the query, n_h is the number of attention heads in the lightweight model, q_t is the t-th token of q, and $\text{Attn}_h(s, q_t)$ denotes the attention score of q_t attending to the privacy span s. Higher w_s indicates that privacy span s receives more attention from other tokens in the user query q, reflecting greater importance.

H Proof of Theorem 5.1

As shown in Figure 1, during the inference stage, only the meta vector and the incomplete query with privacy spans removed are transmitted from the client to the server. The incomplete query does not contain any privacy-sensitive information and is secure for the user. The meta vector contains information about all privacy spans and could be vulnerable to adversaries attempting to reverse-engineer these spans, requiring privacy protection.

PrivacyRestore protects the meta vector by adding noise $\mathcal{N}$ which is sampling from the distribution $p(\mathcal{N}) \propto \exp(-\epsilon\|\mathcal{N}\|)$, before transmission, as shown in Eq 3. Next, we will demonstrate that **injecting noise in this manner adheres to the definition of d_χ-privacy** and effectively protects the user privacy contained in the meta vector.

Assume Z represents the meta vector before adding noise, $\mathcal{R}$ denotes the meta vector after adding noise, as shown in Eq 2 and 3. The process of adding noise can be represented by $\mathcal{M}$. Then, the possibility that Z becomes $\mathcal{R}$ after adding noise $\mathcal{N}$ is

$$\begin{aligned}\mathbb{P}(\mathcal{M}(Z) = \mathcal{R}) &= \mathbb{P}(Z + \mathcal{N} = \mathcal{R}) \\ &= \mathbb{P}(\mathcal{N} = \mathcal{R} - Z) \\ &= \exp(-\epsilon||\mathcal{R} - Z||).\end{aligned} \quad (16)$$

Then for any two meta vectors before adding noise, Z and Z', we have:

$$\begin{aligned}\frac{\mathbb{P}[\mathcal{M}(Z)=\mathcal{R}]}{\mathbb{P}[\mathcal{M}(Z')=\mathcal{R}]} &= \frac{\exp(-\epsilon||\mathcal{R}-Z||)}{\exp(-\epsilon||\mathcal{R}-Z'||)} \\ &= \exp(\epsilon(||\mathcal{R}-Z'|| \\ &\quad - ||\mathcal{R}-Z||)) \\ &\leq \exp(\epsilon||Z'-Z||).\end{aligned} \tag{17}$$

According to the definition of d_χ-privacy in Appendix C.1, the mechanism $\mathcal{M}$ satisfies d_χ-privacy. In other words, by adding noise $\mathcal{N}$, adversaries cannot infer the initial meta vector Z from the meta vector after adding noise $\mathcal{R}$, even if $\mathcal{R}$ is intercepted. Moreover, the privacy budget of our methods is $\epsilon||Z'-Z||$. And considering that Z is the normalization of the weighted sum of restoration vectors, as shown in Eq. 2, then we have:

$$\begin{aligned}\frac{\mathbb{P}[\mathcal{M}(Z)=\mathcal{R}]}{\mathbb{P}[\mathcal{M}(Z')=\mathcal{R}]} &\leq \exp(\epsilon||Z'-Z||) \\ &\leq \exp(2\epsilon)\end{aligned} \tag{18}$$

Thus, the privacy budget of our method is 2ϵ, independent of the input length n and solely depends on the hyperparameter ϵ. In summary, PrivacyRestore fulfills d_χ-privacy and provides a privacy budget 2ϵ which is independent of the input length and inherently addresses the problem of the linear growth of privacy budget.

I Brief Proof of Output Protection

It has been proved by Appendix A of Utpala et al. (2023a) and Section 4.2 of Mattern et al. (2022c) that sampling-based generation can prevent the privacy leakage via the generated output via the Exponential Mechanism. Here, we provide a brief proof that sampling-based generation adheres to the Exponential Mechanism (McSherry and Talwar, 2007), ensuring security for the generated output.

Assume that Q is the user query, $\mathcal{V}$ is the whole token vocabulary, $u \in \mathbb{R}^{|\mathcal{V}|}$ is the output logit, u_t is the logit for the token t in $\mathcal{V}$ and $\mathcal{M}$ denotes the sampling based generation. Recall that, during sampling-based generation, the logit u should be processed by the softmax layer and then be sampled to obtain the output. If T is the sampling temperature and $Pr[\mathcal{M}(Q)=t]$ indicates the probability of generating the token t, then the softmax layer can be expressed by:

$$Pr[\mathcal{M}(Q)=t] = \frac{\exp(u_t/T)}{\sum_{j=1}^{|\mathcal{V}|}\exp(u_j/T)} \tag{19}$$

Let recall the Exponential Mechanism (McSherry and Talwar, 2007), assuming u is the utility function and Δu is the sensitivity of u, then $\mathcal{M}$ satisfy the Exponential Mechanism if and only if

$$\begin{aligned}Pr[\mathcal{M}(Q)=t] &= \frac{\exp(\epsilon u(Q,t)/2\Delta u)}{\sum_{j=1}^{|\mathcal{V}|}\exp(\epsilon u(Q,j)/2\Delta u)} \\ &\propto \exp(\epsilon u(Q,t)/2\Delta u)\end{aligned} \tag{20}$$

By Comparing 19 and 20, we can find that **the sampling from softmax layer follows the definition of Exponential Mechanism**, where $u(Q,t)$ and u_t are different expressions of the same thing. Furthermore, according to the fact that the privacy budget of Exponential Mechanism is ϵ, we can conclude that **the privacy budget of sampling-based generation is** $2\Delta u/T$. The privacy budget decreases with the increasing temperature, indicating that higher temperatures will bring better privacy protection.

J Datasets

Based on the existing benchmarks, such as DDXPlus (Tchango et al., 2022) and NLICE (Al-Ars et al., 2023) for medical diagnosis, and SLJA (Deng et al., 2023) for legal judgment, we construct three privacy-preserving datasets, Pri-DDXPlus, Pri-NLICE and Pri-SLJA, to evaluate the performance of various privacy-preserving methods. In this section, we will introduce the detailed construction process of these three privacy-preserving datasets and provide some statistical information about them.

J.1 Construction Process

The total construction process of these privacy-preserving datasets consists of four stages: **Extraction of Privacy Spans**, **Rewriting Queries for Diversity**, **Assigning Options** and **Filtering Dataset**. The details of these four stage are as follows:

Extraction of Privacy Spans: We used GPT-4 (Achiam et al., 2023) to classify symptoms in DDXPlus and NLICE, as well as case details in SLJA, into five levels ranging from non-sensitive to highly sensitive. The assessment prompt template is shown in Appendix Y.1. A higher level indicates that the symptom or case detail is more sensitive. We define all symptoms and case details with a sensitivity level greater than 3 as privacy spans.

Rewriting Queries for Diversity: The symptom descriptions and case details in the original DDXPlus, NLICE, and SLJA datasets are highly fixed. However, in real-world scenarios, these descriptions are typically more diverse. To address this gap, we utilized GPT-4 (Achiam et al., 2023) to rewrite the user queries in these datasets, ensuring more varied descriptions and differing question formats while preserving the original meaning of the queries. The rewrite prompt template is provided in Appendix Y.2. Here, we provide a rewrite example that demonstrates how rewriting the user query significantly increases the diversity of query descriptions, as shown in Figure 5.

[Queries Before Rewriting]
I have a history of antipsychotic medication usage, nausea, stimulant drug use. I presents the symptoms of involuntary eye movement, jaw pain, muscle spasms, muscle spasms in neck, ptosis, shortness of breath. What is the likely diagnosis?

[Queries After Rewriting] (Varying Format)
I've been taking antipsychotic medication for a while now. I've been feeling really nauseous, and I also used a stimulant drug recently. Lately, I've been having these strange symptoms like my eyes moving involuntarily, and my jaw hurts a lot. I've also been getting muscle spasms in my neck and elsewhere, and my eyelids droop sometimes. On top of that, I've been feeling short of breath. What could be causing all of these symptoms?

Figure 5: A rewrite example displays the diversity enhancement in medical queries. Text highlighted with green background indicates medical history, while yellow background denotes symptoms.

Assigning Options: To evaluate the performance of different privacy-preserving methods, we assign each sample a correct answer along with three randomly selected incorrect options. For DDXPlus and NLICE, we randomly select three diagnosis results to combine with the correct diagnosis as the choices. In the SLJA dataset, we randomly select three legal judgments to pair with the correct one as the options.

Filtering Dataset: The initial dataset is extensive, and we observed that for most samples, removing all privacy spans often yields outputs similar to those obtained when privacy spans are provided. Privacy preserving for these samples is meaningless because users can directly hide those privacy spans and obtain approximate result outputs. In real-world scenarios, sensitive privacy spans often play a crucial role in medical diagnoses and legal judgments, making privacy preservation highly valuable. Our dataset is designed to benchmark various privacy-preserving methods and must include samples where privacy spans are crucial for generating outputs. We utilize the KL divergences to measure the importance scores of samples. We calculate the KL divergence between the model output distributions with and without the privacy symptoms included. A higher KL divergence indicates that the absence of sensitive privacy spans may lead to different or incorrect outputs. We selected only samples with high KL divergence to construct the privacy-preserving datasets. As a result, we curated three privacy-preserving datasets: Pri-DDXPlus and Pri-NLICE for medical diagnosis, and Pri-SLJA for legal judgment.

J.2 Statistical Information

We show the statistics of the obtained Pri-DDXPlus, Pri-NLICE and Pri-SLJA datasets in Table 5. We tally the number of user queries, privacy span types, and privacy spans count. In Pri-DDXPlus and Pri-NLICE, the privacy spans are the symptoms, and the answers are the diagnoses. In Pri-SLJA, the privacy spans are the case details, and the answers are the legal judgments.

Pri-DDXPlus commonly contains more sample instances and more privacy span types compared to Pri-NLICE and Pri-SLJA.

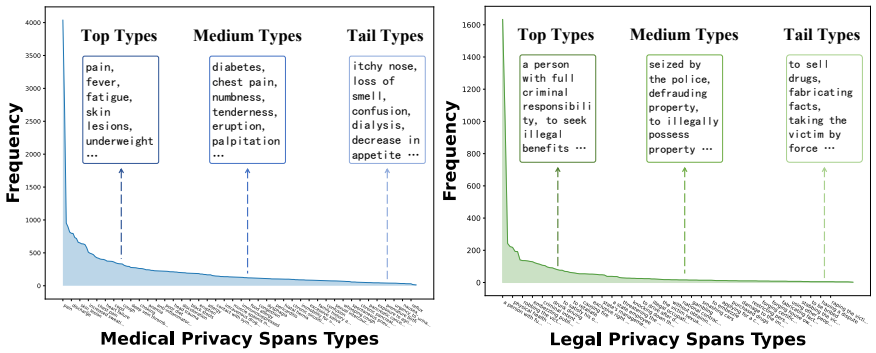
K Long-Tailed Distribution of Privacy Spans

In this section, we present the long-tail distribution of privacy spans, where most privacy spans are concentrated in the majority categories. Here, a privacy span refers to **a specific description of a user's private information**, such as the description of symptoms, e.g., "*I've been having a persistent cough*". The corresponding privacy span type indicates **the category of the private information**, such as the symptom type, e.g., "*cough*".

Considering that we have three privacy-preserving datasets covering the medical and legal domains, we analyze the frequency of each privacy span type separately for each domain. **For the medical domain**, we plot the distribution of medical privacy spans in the Pri-DDXPlus and Pri-NLICE medical dataset, as shown in Figure 6(a). We observe that most medical privacy spans are concentrated on the top types, such as "*pain*" and "*fever*". **For the legal domain**, we plot the distribution of legal privacy spans in Pri-SLJA legal dataset, as shown in Figure 6(b). We also observe

Datasets	Dataset Split	User inputs	Privacy Span Type	Privacy Spans Count
Pri-DDXPlus	All	7759	149	46179
	Train	5901	149	35583
	Dev	309	60	1659
	Test	1549	78	8937
Pri-NLICE	All	4062	64	18241
	Train	3282	64	14933
	Dev	130	58	552
	Test	650	64	2756
Pri-SLJA	All	3901	142	10418
	Train	3117	142	7980
	Dev	130	95	417
	Test	654	142	2021

Table 5: The statistics of Pri-DDXPlus, Pri-NLICE and Pri-SLJA. Average privacy symptoms indicate the average privacy spans occur in one query.



(a) Medical Privacy Spans Distribution (b) Legal Privacy Spans Distribution

Figure 6: Frequency distribution of privacy spans, highlighting the long-tail distribution where a small number of categories dominate the majority of occurrences.

that most legal privacy spans are concentrated on the top types, such as "*a person with full criminal responsibility*".

Therefore, the privacy spans in both the medical and legal domains exhibit a long-tailed distribution, indicating that most privacy spans are concentrated in the majority types.

L Experimental Setup Details

L.1 Evaluation Metrics

To fully evaluate the performance of different privacy-preserving methods, we focus on two aspects: inference performance and inference efficiency. We use **MC1, MC2, ROUGE-L, and LLM-J** to assess inference performance, and **Throughput (TP)** to evaluate inference efficiency. The details of these metrics and their calculation methods are introduced as follows:

MC1/MC2: We employ MC1 and MC2 [3] (Zhang et al., 2024) to measure the model's accuracy in selecting the correct answer among 4 options. We assign each sample in Pri-DDXPlus, Pri-NLICE, and Pri-SLJA with four options, including one correct answer and three incorrect ones. The details of the calculation process are as follows:

As for calculating MC1: For each user input, we select the option with the highest probability as the model's choice. MC1 is defined as the model's accuracy, which is calculated as the proportion of correctly answered inputs.

As for calculating MC2: For each user input, we compute the normalized probability of the correct answer among the four options. The average of these normalized probabilities across all inputs is calculated as the MC2 score.

ROUGE-L: We utilize ROUGE-L (Lin, 2004) to assess the generation ability of different privacy-preserving methods. ROUGE-L primarily mea-

[3]The code is available at https://github.com/sylinrl/TruthfulQA

sures the n-gram overlap between the reference text and the generated text. To evaluate the performance of these privacy-preserving methods, the reference text is the initial output without any privacy protection from the backbone LLM, while the generated text is the output with privacy protection.

LLM-Judge (LLM-J): As ROUGE-L primarily focuses on n-gram overlap between generated text and reference texts, which may not fully capture the semantic meaning or overall quality of the generated content, we further use the LLM-Judge (LLM-J)(Zheng et al., 2023a) metric to assess the generation ability. Specifically, we use the advanced LLM (i.e., GPT-4 (OpenAI, 2023)) to assess the quality of outputs considering relevance, clarity, and accuracy. The assessment prompt is shown in Appendix Y.4. The LLM-J score ranges from 1 to 10, with higher scores indicating better quality.

Throughput (TP): For inference efficiency, we use Throughput (TP), defined as the number of tokens generated per second, to evaluate the inference efficiency. To ensure a fair comparison between different methods, we uniformly use sampling-based generation, as it effectively prevents privacy leakage from the generated outputs, as shown in Appendix I. We set the sampling temperature to 1.0 and the maximum generation length to 1024.

L.2 Compared Methods

Here, we provide a more detailed introduction to all the compared methods, used to protect user privacy during LLM inference, including No Restoration, No Protection, d_χ-privacy (Feyisetan et al., 2020), d_χ-privacy on privacy spans, and Paraphrase (Mattern et al., 2022b; Utpala et al., 2023b). The details are as follows:

No Restoration (lower bound): This method involves transmitting user queries with privacy-sensitive spans removed, without attempting to restore the missing content on the server. As a result, this method serves as the performance lower bound among all privacy-preserving approaches. The degraded quality of the responses highlights the need for effective restoration techniques to bridge the gap between privacy protection and utility.

No Protection (upper bound): In this method, user queries are transmitted directly to the server without any privacy protection or modifications. Since no information is removed or altered, the model operates on fully intact queries, achieving the best possible performance. Consequently, this method establishes the upper bound for all privacy-preserving techniques. The ROUGE-L score for No Protection is always 100.00, as the reference outputs for evaluation are from this method.

d_χ-privacy: As proposed by Feyisetan et al. (2020), we can directly apply d_χ-privacy mechanism to all tokens in the user query by injecting noise into the tokens' embeddings and replacing the initial tokens with their nearest counterparts. This prevents attackers from recovering the original tokens, thereby protecting privacy.

d_χ-privacy on privacy spans: Instead of applying d_χ-privacy mechanism to the entire input, the client can only employ d_χ-privacy only to the privacy spans in the user query, as the other parts of the query contain no privacy-sensitive information. This approach allows for a more appropriate and concise allocation of the privacy budget.

Paraphrase: According to Mattern et al. (2022b); Utpala et al. (2023b), the above methods, both applying d_χ-privacy mechanism to tokens and achieving word-level privatization, suffer from the linear growth problem of the privacy budget. They proposed to use generative models to paraphrase original inputs and achieve privacy protection similar to d_χ-privacy. Due to the client's computational resource limitations and to ensure a fair comparison with our method, we use the FLAN-T5-Base model (Chung et al., 2024) on the client side for paraphrasing in the Paraphrase baseline, as its model size is comparable to that of BERT-Base, which is used in our method.

L.3 Implementation Details

We use Llama2-chat-7b (Touvron et al., 2023) as the LLM backbone on the server side, and BERT-base (Devlin et al., 2019) on the client side for weight estimation, as described in Section 4.2.

During restoration vector training, the LLM parameters remain fixed, and we train the restoration vectors for 5 epochs with a batch size of 1. The optimal number of edited heads K is 175 for Pri-DDXPlus/Pri-SLJA and 125 for Pri-NLICE. The search process is shown in Section U.1.

During generation, we use a sampling-based decoding strategy with a temperature of 1.0 and a maximum generation length of 1024. This is because sampling-based generation can effectively prevent privacy leakage from the generated outputs,

as shown in Appendix I. To evaluate the generation capabilities, we utilize GPT-4 (OpenAI, 2023) to assess the generated outputs. The prompts are detailed in Appendix Y.3.

For the paraphrase baseline method, we employ the flan-t5-base model (Chung et al., 2024) on the client side, as its model size is comparable to BERT-base. Following Mattern et al. (2022b), we clip the final output logits between 0 and 1 during paraphrasing. As a result, the privacy budget for paraphrasing becomes $2n/\tau$, where n represents the maximum length of the user query and τ is the generation temperature.

M Settings of Privacy Hyperparameter ϵ

According to Feyisetan et al. (2019) and the definition of d_χ-privacy in Appendix C.1, when applying the d_χ-privacy mechanism to protect a single token, the privacy budget is ϵd_e and d_e is the maximum distance between any two token embeddings. As proposed by Mattern et al. (2022b), with the length of input text increases, the privacy budget of d_χ-privacy mechanism also grows linearly. Then, assuming that the the maximum length of the user query is n and the maximum length of privacy spans in the user query is n_{sp}, **the privacy budget of d_χ-privacy is** $n\epsilon$ and **the privacy budget of d_χ-privacy on privacy spans is** $n_{sp}\epsilon$. In addition, as pointed by Mattern et al. (2022b); Utpala et al. (2023b), **the privacy budget of paraphrase method is** $2n/\tau$, where τ is the generation temperature used during paraphrasing, and n represents the maximum length of user queries.

The privacy budget of PrivacyRestore is 2ϵ, according to Theorem 5.1. To ensure the same privacy budget for a fair comparison, we need to determine the values of different hyperparameters for different methods on different datasets, such as ϵ for d_χ-privacy (on privacy spans), PrivacyRestore and τ for paraphrase.

Firstly, We set the privacy hyperparameter ϵ to 75.00 for PrivacyRestore. Next, we compute the maximum of the users' inputs lengths n, privacy spans lengths n_{ps}, and distances between word embeddings d_e across three privacy-preserving datasets. Then, we calculate the corresponding ϵ for d_χ-privacy (on privacy spans) and τ for paraphrase to **control the overall privacy budget at 150**, as detailed in Table 6.

N Additional Baselines

In addressing the challenge of safeguarding user privacy during LLM inference, recent studies have explored innovative approaches that leverage small language models to either anonymize or substitute private information within user queries. To evaluate the efficacy of our approach, we implemented two baselines from recent studies on the Pri-DDXPlus dataset. (a). **LLM-anonymization**, proposed by Staab et al. (2024), which uses a language model to anonymize text by repeatedly removing personal attributes identified by an adversarial inference model. (b). **IncogniText**, introduced by Frikha et al. (2024), which anonymizes text by iteratively using an adversarial model to identify private attribute inferences and an anonymization model to rewrite the text, misleading potential attackers into predicting incorrect private attribute values while preserving text utility.

As shown in Table 7, our method significantly outperforms LLM-anonymization and IncogniText on Pri-DDXPlus dataset, strongly validating the effectiveness of our approach.

O Details of Privacy Protection Evaluation

In this section, we provide more details on our implementation of embedding inverse attack (Li et al., 2023b; Morris et al., 2023) and attribute inference attack (Li et al., 2022) to evaluate the privacy protection performance of different privacy-preserving baselines and our method. Lower attack performance indicates stronger privacy protection provided by these methods.

Embedding Inverse Attack: As proposed by Li et al. (2023b); Morris et al. (2023), embedding inversion attacks aim to recover user privacy from the embeddings of user inputs. Specifically, a generative model (e.g., GPT-2 model (Radford et al., 2019)) is used to generate the user's private information based on the given embedding. We implement embedding inversion attacks for the privacy-preserving baselines and our method to evaluate their privacy protection performance. The implementation details are as follows:

We use the gpt2-medium model (Radford et al., 2019) as the generative model, employing greedy search during generation and setting the maximum generation length to 256. **For PrivacyRestore**, the client transmits the incomplete user query and the

Datasets	d_χ-privacy			d_χ-privacy on privacy spans			Paraphrase		PrivacyRestore	Privacy Budget
	n	d_e	ϵ	n_{sp}	d_e	ϵ	n	τ	ϵ	
Pri-DDXPlus	106.00	1.64	**0.86**	49.00	1.64	**1.86**	106.00	**1.41**	**75.00**	**150**
Pri-NLICE	72.00	1.39	**1.50**	38.00	1.39	**2.84**	72.00	**2.08**	**75.00**	**150**
Pri-SLJA	193.00	1.45	**0.54**	42.00	1.45	**2.46**	193.00	**0.78**	**75.00**	**150**

Table 6: The settings of privacy hyperparameters for different baselines across all privacy-preserving datasets.

Methods	MC1 ↑	MC2 ↑	ROUGE-L ↑	LLM-J ↑
LLM-anonymization	52.09(↓10.88)	49.94(↓10.25)	25.22(↓2.02)	2.61(↓1.86)
IncogniText	55.26(↓7.71)	53.73(↓6.46)	25.94(↓1.30)	3.85(↓0.62)
PrivacyRestore	**62.97**	**60.19**	**27.24**	**4.47**

Table 7: Comparison of the performance and the inference efficiency between PrivacyRestore, **LLM-anonymization** and **IncogniText** methods across Pri-DDXPlus dataset. The downward arrow in the table indicates the performance gap of two baseline methods compared with PrivacyRestore.

meta vector. The incomplete user query does not contain any user privacy after removing the privacy spans and is secure for the user. The meta vector contains the information of privacy spans and we perform embedding inverse attack on the meta vector. We use a fully-connected layer to transform the meta vector's dimension to the dimension of hidden state of GPT-2 model. Then we directly input the transformed meta vector as the input embedding. We fine-tune the GPT-2 model and the fully connected layer simultaneously, on the training set for 20 epochs, using a learning rate of 1e-5. **For d_χ-privacy (on privacy spans) and paraphrase**, the client only transmits the garbled user query after applying the d_χ-privacy mechanism or paraphrasing. We then perform the embedding inverse attack on the garbled user query to recover the privacy spans. Here, we do not need to transform the dimension and can directly input the garbled user query as the input context for the GPT-2 attack model. Then attack model can recover the privacy spans according to the garble user query. We finetune the GPT-2 model on the training set for 20 epochs using the learning rate of 1e-5.

To evaluate the attack's performance, we compute the ROUGE-L score between the generated output of the attack model and ground true privacy spans in the user query, where higher scores indicate better attack effectiveness.

Attribute Inference Attack: According to Li et al. (2022), attribute inference attack attempts to infer user's private attribute even when the user query is protected by some privacy-preserving methods. In our scenario, we use attribute inference attacks to infer the privacy spans in the user query. The implementation details are as follows:

Following Li et al. (2022), we construct a multilayer perceptron (MLP) as the classifier, with the output dimension corresponding to the entire vocabulary size. We use the classifier to predict the token IDs of the privacy spans in the user query. Since the query contains multiple privacy spans, and each span consists of multiple tokens, this classification task is a multi-label classification. **For PrivacyRestore**, we also perform attribute inference attacks on the meta vector, so the input dimension of the classifier corresponds to the dimension of the meta vector. We finetune the classifier on the training set for 20 epochs using the learning rate of 1e-5. **For d_χ-privacy (on privacy spans) and paraphrase**, we perform attribute inference attack on the garbled user query. We utilize GPT-2 model (Radford et al., 2019) to process the query and obtain the last token's hidden state as the vector representation. Classification is then performed on this hidden state. We finetune the classifier and the GPT-2 model jointly, on the training set for 20 epochs using the learning rate of 1e-5.

To evaluate the attack's performance, we calculate the F1 score of the classification, where a higher F1 score indicates a more successful attack.

P More Privacy Protection Evaluation Results

P.1 Concatenated Text Attack

In Section 6.3, the implementation of embedding inverse attack follows previous work (Li et al., 2023b), which merely takes meta vectors derived from privacy spans as input. This approach, however, may overlook the contextual information in the incomplete user query. Therefore, we propose the Concatenated Text Attack by firstly using embedding inverse attack to transform the meta vector to the text format and then concatenate it with the incomplete user query to add more contextual

information for recovering privacy spans. The implementation details are as follows:

We finetune two attack models: one to transform the meta vector into text format, and the other to recover privacy spans from the concatenated text. For the first model, we finetune a GPT-2 model, where the input is the meta vector and the output is the privacy spans, similar to the embedding inverse attack process. Then we concatenate the generated output from the first model with the incomplete user query as the input to the second model. As for the second model, we also finetune a GPT-2 model which aims to utilized the incomplete user query to improve the quality of the generated output from the first attack model. For both attack models, we finetune them on the train set for 20 epochs using the learning rate of 1e-5. We also utilized the ROUGE-L scores between the recovered results and the privacy spans as the evaluation metric.

The experimental evaluation of the Concatenated Text Attack is presented in Table 8. The experiment results show that, although unifying the vector format of the meta vector and the text format of incomplete user query, the attack performance for our method is still poor, demonstrating the effectiveness of our method.

ϵ values	1	20	40	75	125	175
Pri-DDXplus	0.0112	0.0107	0.0130	0.0093	0.0115	0.0024
Pri-NLICE	0.0566	0.0486	0.0427	0.0467	0.0423	0.0350
Pri-SLJA	0.0027	0.0011	0.0022	0.0024	0.0021	0.0028

Table 8: ROUGE-L Scores for Concatenated Text Attack Across Different ϵ Values

P.2 Simulating Activation Steering Attack

We assume the attacker is aware that the meta vector will be used for activation steering on the server for information restoration. **The attacker can also simulate activation steering while recovering the privacy spans in the user query**. Considering the LLM weights on the server are kept secret, the attack only can conduct the activation steering on the other generative model, such as GPT-2 (Radford et al., 2019) model. The implementation details are as follows:

First, due to the heterogeneity between the attack model (GPT-2) and the LLM on the server (Llama-2-7b), we use a fully connected layer to transform the meta vector's dimension to fit in the attack model. Specifically, since the meta vector is applied to the head output and its initial dimension matches the head output of the LLM, the fully connected layer adjusts it to the head output dimension of the attack model (GPT-2). Next, we input the incomplete user query into the attack model and use the adjusted meta vector to perform activation steering, prompting the model to generate the privacy spans in the query. We fine-tune the GPT-2 model and the fully connected layer jointly for 20 epochs with a learning rate of 1e-4. We utilized the ROUGE-L scores between the recovered results and the privacy spans as the evaluation metric.

As shown in Table 9, the Simulating Activation Steering Attack demonstrated limited performance across various ϵ values on all three datasets. This weakness may be attributed to that the meta vector are trained offline for the server's LLMs. Although we have used fully connected layer to transform the dimension of the meta vector, applying the meta vector to the attack model still leads to incompatibility.

ϵ values	1	20	40	75	125	175
Pri-DDXplus	0.0023	0.0329	0.0321	0.0329	0.0310	0.0365
Pri-NLICE	0.0165	0.0123	0.0118	0.0170	0.0283	0.0315
Pri-SLJA	0.0161	0.0818	0.0862	0.0861	0.1048	0.1059

Table 9: ROUGE-L Scores for Simulating Activation Steering Attack Across Different ϵ Values

P.3 Hidden State Attack

Hidden State Attack(Carlini et al., 2021) attempts to perform a training data extraction attack to recover individual training examples by querying the language model. We have implemented a Hidden State Attack employing the LLaMA-7B architecture as the designated attack model. The objective of this attack was to infer private information from steered hidden states, specifically targeting the first, sixteenth, and final layers of the network. The efficacy of the attack was quantitatively assessed using the ROUGE-L metric, which measures the lexical similarity between the output generated by the attack model and the original user query containing sensitive private information.

As indicated by the empirical results presented in Table10, the attack directed at the final layer demonstrated a marginal improvement in performance when compared to attacks executed on the first and sixteenth layers. This observation can be attributed to the fact that hidden states within the final layer are fully restored, in contrast to the earlier

layers where only partial restoration is achieved. Nevertheless, it is crucial to emphasize that the performance of all conducted attacks remained notably weak. This outcome underscores the robustness of our proposed methodology in safeguarding user privacy. Furthermore, it is pertinent to state that our method does not introduce privacy vulnerabilities at the model level, as noise has been incorporated into the meta vector, thereby obfuscating sensitive information.

ϵ values	0.01	0.86	2.01
Hidden State Attack on 1st layer	0.0023	0.0329	0.0321
Hidden State Attack on 16th layer	0.0165	0.0123	0.0118
Hidden State Attack on final layer	0.0161	0.0818	0.0862

Table 10: ROUGE-L Scores for Hidden State Attack on Different Layers Across Different ϵ

Q Analysis of Output Privacy Protection

In this section, we evaluate the privacy leakage in the generated output of our method by implementing Embedding Inversion Attacks (EIA) and Attribute Inference Attacks (AIA). We also directly count the frequency of privacy span occurrences in the generated outputs. The details of these attack methods are as follows:

Embedding Inverse Attack for the generated output. Embedding inversion attacks (Li et al., 2023b; Morris et al., 2023) directly utilize the generative model (e.g., GPT-2) to generate privacy spans in the user query based on the attacked embedding. Although the generated output is in text format rather than embedding format, we still input it into the GPT-2 model to generate the privacy spans from the user query.

To be specific, we utilize the GPT-2 model (Radford et al., 2019) as the generative model and set the maximum generation length to 256. The input of the GPT-2 attack model is the generated output and the target output is the privacy spans in the user query. We finetune the GPT-2 model on the training set for 20 epochs using the learning rate of 1e-5. To evaluate attack performance, we compute the ROUGE-L score between the output generated by the attack model and the ground truth privacy spans in the user query.

Attribute Inference Attack for the generated output. Attribute inference attack (Li et al., 2022) attempts to steal user privacy by performing classification on the generated output, where the target labels corresponding to the token IDs of those privacy spans. Since each user query contains multiple privacy spans and each privacy span contains multiple tokens, this classification task is naturally a multi-label classification task.

First, we use the GPT-2 model (Radford et al., 2019) to process the text input and obtain the hidden state of the last token as its vector representation. Next, following Li et al. (2022), we construct a multi-layer perceptron (MLP) model as the classifier. The classifier's input is the vector representation, and the output dimension corresponds to the vocabulary size. We finetune the GPT-2 model along with the MLP on the training set for 20 epochs, using a learning rate of 1e-5. To evaluate the attack performance, we compute the F1 score of the classification results, where a higher F1 score indicates a more successful attack.

R Details of Privacy Protection Robustness for Long Queries

In this section, we will provide more implementation details and experiment results analysis when evaluating the privacy protection robustness of d_χ-privacy and our method.

R.1 Different Protected Text Length for d_χ-privacy

As shown in Section 6.4, we randomly select a proportion of token in user query to simulate the protected text and larger proportion indicates longer protect text. The proportion of selected token is denoted as the **d_χ-privacy Percentage**. As presented by Feyisetan et al. (2019, 2020), the d_χ-privacy mechanism protects input by injecting noise into the token embeddings and replacing the original tokens with their nearest neighbors. To attack the garbled query, we implement two types of attacks: prompt injection attack (Suo, 2024) and attribute inference attack (Li et al., 2022), both commonly used for attacking text inputs. The details of implementation of these two attack methods are as follows:

For prompt injection attack, following Suo (2024), we add extra instructions before and after the garbled query, to prompt the LLM in the server to output the protected text instead of following the initial user query. And then we intercept the output returned by the LLM on the server for user privacy. The template for the additional instructions

is provided in Appendix Y.5. To evaluate the attack performance, we calculate the ROUGE-L score between the returned output and the protected text. A higher ROUGE-L score indicates greater overlap between the returned output and the protected text, signifying more successful attack results. **For attribute inference attack**, inspiring by Li et al. (2022), We firstly utilize GPT-2 model (Radford et al., 2019) to process the garbled query and obtain the last token's hidden state. Next, we construct a multi-layer perceptron (MLP) as the classifier to classify the hidden states, with the target labels being the token IDs of the protected text. This is a multi-label classification task. We finetune the classifier and the GPT-2 model on the training set for 20 epochs using the learning rate of 1e-5. The attack performance is evaluated using the classification F1 score.

As shown in Figure 3(a) and Figure 3(b), the attack performance of prompt injection attack and attribute inference attack across all three datasets are all grows with the larger d_χ-privacy percentage. These experiment results reflect the linear growth problem of privacy budget in d_χ-privacy.

R.2 Different Protected Text Length for PrivacyRestore

For PrivacyRestore, we randomly choose a proportion of privacy spans in the user query as the protected text and the proportion is denoted as the **Privacy Span Ratio** α. Larger α indicate the longer protected text. Considering that, in our method, the client only transmits the incomplete query with the meta vector and the incomplete query contains no privacy information, then we implement embedding inverse attack (Li et al., 2023b; Morris et al., 2023) and attribute inference attack (Li et al., 2022) on the meta vector across different α values. The details of implementation of these two attack methods are as follows:

For embedding inverse attack, we firstly fully-connected layer to transform the meta vector's dimension to the dimension of hidden state of GPT-2 attack model. Then we directly input the transformed meta vector as the input embedding to the attack model, prompting it to generate the privacy spans in the user query. We finetune the fully-connected layer with the GPT-2 attack model on the training set for 20 epochs using the learning rate of 1e-5. The attack performance is assess by the ROUGE-L score between the generated output from the attack model and the protected text. **For attribute inference attack**, we construct a multi-layer perceptron (MLP) as the classifier to classify the meta vector, with the target labels being the token IDs of the protected text. This is also a multi-label classification task. We finetune the classifier on the training set for 20 epochs using the learning rate of 1e-5. The attack performance is evaluated using the classification F1 score.

As shown in Figure 3(c) and 3(d), the ROUGE-L score for the embedding inverse attack remains nearly stable across different α values in the Pri-SLJA and Pri-DDXPlus datasets. What's a little strange is the ROUGE-L score in the Pri-NLICE dataset shows a slight increase. The possible reason is that higher ratio indicating more privacy spans and **resulting longer reference string when compute the ROUGE-L score**. Since ROUGE-L measures the overlap between the generated output and the reference string, a longer reference string may slightly boost the score. The F1 score for the attribute inference attack remains stable across all three datasets. The stable performance in both attack scenarios provides empirical support for Theorem 5.1. Our method effectively and inherently solves the linear growth problem of the privacy budget, achieving robust and stable privacy protection performance regardless of the length of the protected text, even with long protected text.

S Details of Evaluation of Handling Out-of-Set Privacy Spans

In this section, we will evaluate our method when handling those out-of-set privacy spans. As shown in Figure 6, most of privacy spans focus on the majority categories. Our core set of predefined privacy spans easily covers the majority of categories, even though it cannot cover all privacy span types. To evaluate the performance of our method when the core set cannot cover all privacy span types, we assume that the core set contains only the top 5, 40, 80, 100, or 120 privacy span types and assess our method. Additionally, we provide results when the core set covers all 149 privacy span types in the Pri-DDXPlus dataset.

As shown in Table 11, our approach outperforms the No Restoration baseline, with performance gains increasing as the predefined span set expands. Notably, **even when limited to the top 100 types, our method achieves significant improvements across multiple metrics**. These findings highlight the robustness and efficiency of our method in han-

Methods↑	MC1↑	MC2↑	RL↑	LLM-J↑
No Restoration (lower bound)	33.57	32.49	25.19	3.21
Predefine only top 5	38.21	36.17	25.82	3.57
Predefine only top 40	44.28	42.00	25.89	3.83
Predefine only top 80	45.83	43.53	26.59	3.95
Predefine only top 100	**54.93**(↑ 21.36)	**52.15**(↑ 19.66)	**26.37**(↑ 1.18)	**4.19**(↑ 0.98)
Predefine only top 120	58.42	55.40	26.87	4.27
Predefine only all (top 149)	62.97(↑ 29.40)	60.19(↑ 27.70)	27.24(↑ 2.05)	4.47(↑ 1.26)

Table 11: Performance comparison across different predefined privacy span type sets $\mathcal{C}$ in Pri-DDXPlus

dling those out-of-set privacy spans when our predefined cores set cannot cover all privacy spans.

T Details of Extension for Users Unable to Determine Privacy Spans

In this section, we evaluate the performance of combining PrivacyRestore with existing text sanitization techniques (Kan et al., 2023; Chen et al., 2023a) to address the situation where users cannot or are unwilling to determine privacy spans themselves. In our main setting, we follow the principle of "Information Self-Determination Right" and assume that the user should determine the privacy spans in their queries by themselves. However, we also consider the situation when the user cannot or is unwilling to identify the privacy spans. Thanks to our method is totally orthogonal to the existing Text Sanitization techniques (Kan et al., 2023; Chen et al., 2023a), we can use text sanitization technique to identify and remove privacy spans automatically and restore information during LLM inference by our method.

Specifically, the pipeline of combining text sanitization technique and our method consists of three stages: **Privacy Spans Identification**, **User Query Sanitization** and **PrivacyRestore**. The details of these three stages are as follows:

Privacy Spans Identification: Following Kan et al. (2023); Chen et al. (2023a), we construct a classifier based on the BERT-base-uncased model (Devlin et al., 2019). The input to the classifier is the user query, and the target labels are the types of privacy spans in the query. Considering that each query contain multiple privacy spans and this is a multi-label classification task. We use the classifier to identify the types of privacy spans present in the user query. We finetune the classifier on the training set for 10 epochs using the learning rate of 1e-4. To evaluate the identification performance, we compute the precision, recall and F1 score of the classification.

As shown in Table 12, the classification results of our classifier are superior, achieving an F1 score of 99.66.

	Precision	Recall	F1
Privacy Spans Identification	$99.16_{\pm 0.21}$	$98.78_{\pm 0.31}$	$99.66_{\pm 0.27}$

Table 12: Privacy Spans Identification accuracy. The results of the three experiments are presented, with the variance displayed in subscript.

User Query Sanitization: After identifying all privacy spans in the user query, we need to remove all these privacy spans from the user query to achieve sanitization. Inspiring by Kan et al. (2023); Chen et al. (2023a), we finetune a Qwen-2.5-0.5B model (Yang et al., 2024) to conduct the text sanitization. Specifically, the model takes the user query and the identified privacy span types as input and outputs a sanitized version of the user query with the privacy spans removed. We finetune the Qwen-2.5-0.5B model on the train set for 15 epochs using the learning rate of 1e-5.

To evaluate the efficacy of the text sanitization, we conducted both Attribute Inference Attacks (AIA) and Embedding Inversion Attacks (EIA) on the sanitized queries. As shown in Table 13, the performance of both attack methods are very low, demonstrating that our sanitization method can effectively protect the user privacy.

	EIA (ROUGE-L)	AIA (F1)
No Protection	0.40	0.70
Sanitized Results	0.06(↓ 0.34)	0.07(↓ 0.63)

Table 13: Attack results on sanitized queries. EIA refers to the embedding inverse attack, with the evaluation metric being ROUGE-L. AIA denotes the attribute inference attack, evaluated using the F1 score.

PrivacyRestore: Following the text sanitization, we use PrivacyRestore to restore the information during LLM inference on the server. We present the

	MC1↑	MC2↑	RL↑	LLM-J↑
No Restoration	33.57	32.49	25.19	3.21
TS only	29.63(↓ 3.94)	30.85(↓ 1.64)	25.45(↑ 0.26)	3.46(↑ 0.25)
PR+PS	62.97(↑ 29.40)	60.19(↑ 27.70)	27.24(↑ 2.05)	4.47(↑ 1.26)
PR+TS	**62.87**(↑ 29.30)	**59.97**(↑ 27.48)	**26.47**(↑ 1.28)	**4.28**(↑ 1.07)

Table 14: The performance of combining our method with text sanitization technique. **TS only** indicates only use sanitization methods without combining PrivacyRestore. **PR+PS** indicates PrivacyRestore when the user can determine privacy spans by themselves. **PR+TS** denotes combining PrivacyRestore and text sanitization to address the situation when the user cannot identify privacy spans by themselves. Three methods are compared with **No Restoration** baseline (lower bound).

performance results of our method when user can determine privacy spans (**PR+PR**), combining our method with text sanitization (**PR+TS**), only using text sanitization (**TS only**) and the No Restoration baseline in Table 14.

As the experiment results show, even in scenarios where users are unable to identify privacy spans, the combination of our method with text sanitization (PR+TS) results in a significant enhancement in performance compared to the No restoration baseline (lower bound) and only using text sanitization (only TS). The utility performance achieved is notably superior, suggesting that our method is effective in preserving privacy while simultaneously optimizing utility. Moreover, the performance metrics of combining our method with text sanitization are comparable to those when the user can determine privacy spans themselves (PR+PS). This comparison further underscores the robustness of combining our method with text sanitization and validates the efficacy of our approach in real-world applications, even when users cannot determine privacy spans themselves.

U Details to Ablation Study

In this section, we conduct additional experiments to analyze the impact of the number of edited heads and evaluate the performance of our method across varying LLM backbones.

U.1 Hyperparameter Analysis of the Number of Edited Heads

We evaluate the performance of our methods using different numbers of edited heads, K, across the development sets of three privacy-preserving datasets. For simplicity, we compute MC2 to represent classification performance, LLM-J to measure generation performance, and TP to indicate inference efficiency.

As shown in Table 15, according to the MC2 score, the optimal value of K is 175 for the Pri-DDXPlus and Pri-SLJA datasets, and 125 for the Pri-NLICE dataset. The performance degradation as K increases can be attributed to the cumulative effect of multiple edited heads. As more heads are modified, the activations progressively deviate from their initial values, potentially compromising the LLM's general capabilities. Moreover, throughput increases with larger K because we need to inject the meta vector for each head in $\mathcal{H}_k$ using Eq 4 on the server. Consequently, more heads indicate more injections, which increases the inference time on the server.

U.2 Varying LLM Backbone

We evaluate the performance of PrivacyRestore and other privacy-preserving baselines on a larger model, Llama-13b-chat.

As shown in Figure 7, PrivacyRestore outperforms the other baselines in terms of both MC2 and LLM-J values across all three privacy-preserving datasets. Notably, the performance of all privacy-preserving methods on the larger model, Llama-13b-chat, is worse than on the smaller model, Llama-7b-chat. This suggests that as model size increases, the model becomes more sensitive to the injected disturbances introduced by these privacy-preserving methods, leading to performance degradation.

V Inference Efficiency Analysis

In this section, we provide a detailed analysis of the computational efficiency of our proposed method, specifically addressing its performance in terms of training time and inference throughput. Concerns regarding the computing resource requirements and time costs associated with training recovery vectors, building meta vectors, and other operations in practical applications have been noted. To address these, we have conducted a series of experiments to evaluate the efficiency of our approach, with a particular focus on its suitability for large-scale deployments.

Table 16 presents a comprehensive overview of the performance metrics. The evaluation was conducted on three distinct datasets: Pri-DDXplus, Pri-NLICE, and Pri-SLJA. We report the total training time, the number of trainable parameters specific to

Datasets	Metrics	$K = 75$	$K = 100$	$K = 125$	$K = 150$	$K = 175$	$K = 200$
Pri-DDXplus	MC2 ↑	52.20	56.17	59.39	58.96	**62.95**	62.64
	LLM-J ↑	4.51	4.38	4.45	4.33	**4.71**	4.55
	TP ↑	**24.31**	21.51	19.72	20.07	22.68	21.91
Pri-NLICE	MC2 ↑	37.15	51.01	**58.97**	51.89	58.11	58.45
	LLM-J ↑	3.27	3.66	**3.80**	3.44	3.40	3.62
	TP ↑	**20.05**	19.14	18.23	16.08	15.89	15.48
Pri-SLJA	MC2 ↑	28.75	30.65	35.07	32.41	**35.13**	32.08
	LLM-J ↑	5.21	**5.41**	5.00	5.33	5.15	5.28
	TP ↑	**36.28**	35.25	34.62	32.97	30.51	29.87

Table 15: The performance of PrivacyRestore on the development set using various numbers of edited heads K. MC2 reflects classification capability, while LLM-J indicates generation performance. The TP assesses inference efficiency. We report results across three datasets to identify the optimal K for each datasets. The best results are highlighted in bold.

Dataset	Train Time	PrivacyRestore Params	Params Ratio	PrivacyRestore TP	Initial TP	TP Ratio
Pri-DDXplus	8h	26M	4%	26.09	41.08	64%
Pri-NLICE	7h	8M	1%	32.33	41.44	78%
Pri-SLJA	10h	26M	4%	30.73	39.49	77%

Table 16: Experimental Results on Inference Efficiency of PrivacyRestore under different datasets. The table shows training time, trainable parameters (and their ratio to full model), inference throughput of our method, initial throughput, and the ratio of our method's throughput to the initial one.

our method, the ratio of these trainable parameters to the full model parameters, the inference throughput achieved using our method, the initial inference throughput (baseline), and the percentage of our method's inference throughput relative to the initial throughput.

The experimental results demonstrate that our method maintains a low computational overhead during the training phase. For instance, the training times for Pri-DDXplus, Pri-NLICE, and Pri-SLJA were 8 hours, 7 hours, and 10 hours, respectively. These durations are considered acceptable, particularly given that our approach focuses on training only the restoration vectors. Crucially, these restoration vectors constitute a small fraction of the total model parameters, ranging from just 1% (for Pri-NLICE with 8M trainable parameters) to 4% (for Pri-DDXplus and Pri-SLJA with 26M trainable parameters). This targeted training strategy significantly reduces the computational burden compared to retraining an entire model, making it highly efficient.

In the inference stage, our method demonstrates commendable performance by retaining a substantial portion of the original inference throughput. Specifically, the inference throughput achieved by our method was 26.09 for Pri-DDXplus, 32.33 for Pri-NLICE, and 30.73 for Pri-SLJA. When compared to the initial inference throughputs of 41.08, 41.44, and 39.49, respectively, our method sustains between 64% and 78% of the original throughput. This indicates that while introducing the restoration mechanism, the impact on inference speed is managed effectively. For example, with Pri-NLICE, our method achieved 78% of the initial throughput while only requiring the training of 1% of the model parameters. Similarly, for Pri-SLJA and Pri-DDXplus, we achieved 77% and 64% of the initial throughput, respectively. Furthermore, the method retains a high percentage (65%-80%) of the original inference throughput, indicating minimal overhead during the inference phase. These characteristics collectively demonstrate that our approach is not only effective but also computationally efficient, rendering it well-suited and feasible for deployment in large-scale applications where both training and inference costs are critical considerations. The results affirm that the method remains low-cost and efficient across both training and inference stages.

Percentage of Spans Removal	MC1	MC2	LLM-J
No More Spans Removed	62.97	60.19	27.24
1% More Spans Removed	62.13	59.51	27.21
5% More Spans Removed	59.13	55.05	27.16
10% More Spans Removed	57.45	54.85	26.51
30% More Spans Removed	50.87	48.43	25.42
50% More Spans Removed	42.47	41.73	23.90

Table 17: Performance of PrivacyRestore under varying percentages of additional span removal. The table displays the MC1, MC2, and LLM-J scores across varying percentages of removed non-privacy spans.

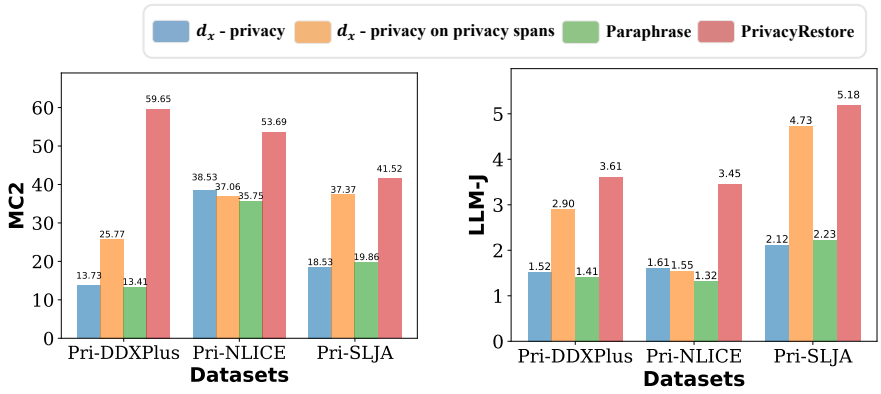
W Example Outputs of PrivacyRestore

We provide some example outputs of our method in Figure 8. As shown in these examples, applying d_χ-privacy to privacy spans results in outputs with higher ROUGE-L scores but lower LLM-J scores compared to our method. After analyzing these outputs in detail, the high ROUGE-L scores from d_χ-privacy on privacy spans likely result from a greater overlap with the initial output. However, the overlapping sections consist mainly of meaningless sentence structures and lack diagnostic information. Moreover, the final diagnosis is incorrect, leading to lower LLM-J scores. In contrast, PrivacyRestore generates outputs with a different structure but provides the same, correct diagnosis. As a result, our method achieves slightly lower ROUGE-L scores but significantly higher LLM-J scores compared to d_χ-privacy on privacy spans.

X Privacy Spans Over-Removal

To highlight the efficiency of PrivacyRestore in mitigating the impact of user errors during privacy span removal, we conducted a series of experiments to evaluate its robustness under adverse conditions. Recognizing that users may inadvertently remove longer spans than necessary, our study simulated scenarios where, in addition to the essential privacy spans, an extra 1%, 5%, 10%, 20%, 30%, and even 50% of non-privacy text was removed from the Pri-DDXPlus dataset.

Shown in Table 17, although removing longer spans than necessary can impact performance, the degradation is minimal. Even with an extra 30% of spans removed, our method still achieves robust scores—50.87 in MC1, 48.43 in MC2, 25.42 in ROUGE-L, and 3.65 in LLM-J.



(a) MC scores

(b) LLM-J scores

Figure 7: The MC2 and LLM-J results of PrivacyRestore and other privacy-preserving baselines on larger model, Llama-13b-chat, across three datasets.

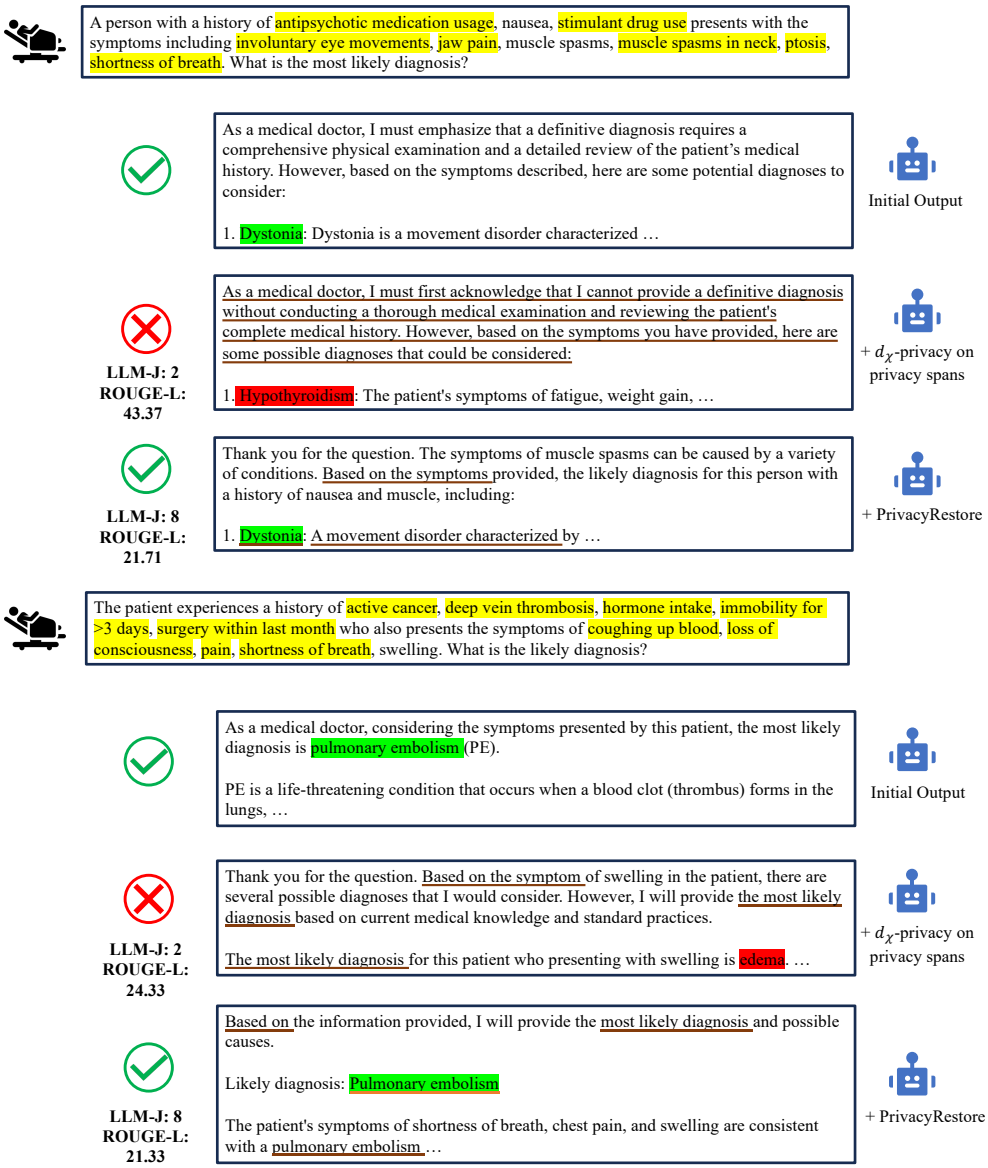


Figure 8: Example Outputs of PrivacyRestore and d_χ on privacy spans in Pri-DDXPlus dataset. Text highlighted with a yellow background represents the privacy spans in user inputs. Text highlighted with a green background indicates the correct diagnosis. Text highlighted with a red background denotes the incorrect diagnosis. Underscored text marks sections that overlap with the initial output.

Y Prompt Template Details

Y.1 Classification of Privacy Spans.

Y.1.1 Medical Datasets (Pri-DDXPlus/Pri-NLICE).

Prompt template shown in Figure 9 is for GPT and is used to classify symptoms in Pri-DDXPlus/Pri-NLICE dataset into sensitive and non-sensitive categories. GPT grades the symptoms on a scale of one to five based on sensitivity, with levels greater than three considered private spans in the Pri-DDXPlus/Pri-NLICE dataset.

Y.1.2 Legal Dataset (Pri-SLJA).

Prompt template shown in Figure 10 is for GPT and is used to classify the case details in Pri-SLJA dataset into sensitive and non-sensitive categories. GPT grades the symptoms on a scale of one to five based on sensitivity, with levels greater than three considered private spans in the Pri-SLJA dataset.

Y.2 Rewriting of User Queries.

Y.2.1 Medical Datasets (Pri-DDXPlus/Pri-NLICE).

The prompt template shown in Figure 11 is designed for GPT and is utilized to rewrite medical queries in the Pri-DDXPlus and Pri-NLICE datasets.

Y.2.2 Legal Dataset (Pri-SLJA).

The prompt template shown in Figure 12 is designed for GPT and is utilized to rewrite medical queries in the Pri-SLJA dataset.

Y.3 Generation Prompts.

Y.3.1 Medical Datasets (Pri-DDXPlus/Pri-NLICE).

Prompt template shown in Figure 13 is for Llama model and is used during model generation for the Pri-DDXPlus/Pri-NLICE datasets.

Y.3.2 Legal Dataset (Pri-SLJA).

Prompt template shown in Figure 14 is for Llama model and is used during model generation for the Pri-SLJA datasets.

Y.4 Evaluation of Generated Output.

Y.4.1 Medical Datasets (Pri-DDXPlus/Pri-NLICE).

Prompt template shown in Figure 15 is for GPT and evaluates the quality of generated output based on relevance, clarity, and accuracy, for Pri-DDXPlus/Pri-NLICE dataset. Scores range from 1 to 10, with higher values indicating better output.

Y.4.2 Legal Dataset (Pri-SLJA).

Prompt template shown in Figure 16 is for GPT and evaluates the quality of generated output based on relevance, clarity, and accuracy, for Pri-SLJA dataset. Scores range from 1 to 10, with higher values indicating better output.

Y.5 Prompt Injection Attack.

Prompt template shown in Figure 17 is for Llama model and is used to carry out a prompt injection attack, translating the garbled text back into the original text.

You are a privacy expert with extensive medical knowledge. Given a potential symptom or antecedent, please categorize symptom or antecedent into five grades based on the level of privacy they entail.

Utilize the classification guidelines provided below to categorize the symptom or antecedent:
Degrade 1: public information, symptoms that are common, widely known, and do not reveal any personal or sensitive information. Examples include sneezing, headache, or minor injuries.
Degrade 2: non-sensitive personal information, symptoms that may be personal but not necessarily sensitive or revealing. These may include common illnesses like cold or flu, allergies, or minor digestive issues.
Degrade 3: potentially sensitive information, symptoms that could be indicative of underlying health conditions but are not immediately sensitive or stigmatizing. Examples include chronic conditions like diabetes, hypertension, or asthma.
Degrade 4: sensitive personal information, symptoms that may be stigmatizing or have social implications if disclosed publicly. This could include mental health issues like depression or anxiety, reproductive health concerns, or substance abuse.
Degrade 5: highly sensitive information, symptoms that are highly personal, stigmatizing, or potentially life-altering if disclosed publicly. This category includes sexually transmitted infections, HIV/AIDS, certain types of cancer, or rare and serious medical condition.
Output your evaluation in the following format:

#thereason: your analysis here [note: this should be step-by-step analysis following the steps: (a) Give an overview of the given symptom or antecedent. (b) Evaluate the degree of the given symptom or antecedent according to classification guidelines.]
#thescore: your degrade here. [note: provide your degrade in based on your analysis and the classification criteria. Please note that only the degrade is required to be written in here]
Symptom: **[INSERT YOUR SYMPTOMS HERE].**

Figure 9: Prompt template for classifying symptoms

You are a privacy expert with extensive legal knowledge, please categorize the given case element into five levels based on the degree of privacy involved.

Utilize the classification guidelines provided below to categorize the case element:
Degrade 1: public legal information: Information that is publicly available and does not reveal personal or sensitive legal details. Examples include general legal concepts, publicly available case laws, or minor regulatory matters.
Degrade 2: non-sensitive personal legal information: Legal information that may involve a specific individual but is not particularly sensitive or revealing. Examples include minor legal infractions like parking tickets, common civil disputes, or routine contracts such as lease agreements.
Degrade 3: potentially sensitive legal information: Legal information that could imply more personal aspects but is not immediately sensitive or damaging. Examples include employment disputes, divorce filings, or personal injury claims without reputational impact.
Degrade 4: sensitive personal legal information: Information that could be stigmatizing or have significant social, financial, or reputational consequences. Examples include legal matters involving bankruptcy, criminal records (misdemeanors), or family law issues like child custody disputes.
Degrade 5: highly sensitive legal information: Legal details that are highly personal, stigmatizing, or potentially life-altering if disclosed. This includes serious criminal cases (e.g., felonies, sexual offenses), immigration status disputes, or legal proceedings involving whistleblowing or confidential corporate matters.
Output your evaluation in the following format:
#thereason: your analysis here [note: this should be step-by-step analysis following the steps: (a) Give an overview of the given case element. (b) Evaluate the degree of the given case element according to classification guidelines.]
#thescore: your degrade here. [note: provide your degrade in based on your analysis and the classification criteria. Please note that only the degrade is required to be written in here]
Case element: **[INSERT CASE ELEMENTS HERE].**

Figure 10: Prompt template for classifying case details

[INST] <<SYS>>
You are a language model specialized in the medical domain. You will be given a simple, fixed-format description of a patient's symptoms or medical query. Your task is to rephrase this description into a more detailed, realistic, and complex version that reflects how patients typically describe their conditions to healthcare professionals. Feel free to add any relevant details that may be implied or missing, such as symptom duration, severity, or additional context that could aid in a more accurate assessment. Additionally, ensure that the transformed query is presented in a variety of formats, using different phrasing, structures, and levels of specificity to reflect natural variations in how patients might express their concerns. The output should sound realistic and diverse, while staying true to the initial medical context.
<</SYS>>

[INSERT USER INPUT HERE] [/INST]

Figure 11: Prompt template for rewriting queries in medical datasets

[INST] <<SYS>>
You are a language model specialized in the legal domain. You will be given a simple, fixed-format description of a legal fact pattern or query. Your task is to rephrase this description into a more detailed, realistic, and nuanced version that reflects how individuals or legal professionals typically articulate such situations in legal consultations or documents. Feel free to add any relevant details that may be implied or missing, such as background context, the sequence of events, or potential legal implications that could provide a more comprehensive understanding. Additionally, ensure that the transformed query is presented in a variety of formats, using different phrasing, structures, and levels of specificity to reflect natural variations in how legal issues might be expressed. The output should sound realistic and diverse while staying true to the initial legal context.
<</SYS>>

[INSERT USER INPUT HERE] [/INST]

Figure 12: Prompt template for rewriting queries in legal datasets

[INST] <<SYS>>
You are a medical doctor answering real-world medical entrance exam questions. Based on your understanding of basic and clinical science, medical knowledge, and mechanisms underlying health, disease, patient care, and modes of therapy, answer the following diagnosis question. Base your answer on the current and standard practices referenced in medical guidelines.
<</SYS>>

[INSERT USER INPUT HERE] [/INST]

Figure 13: Pri-DDXPlus/Pri-NLICE prompt template for model generation

[INST] <<SYS>>
You are the world's best legal judgment machine, tasked with rendering a verdict in a legal case. Below are four paragraphs describing the case, though some information may be missing.
Subject refers to the individual with criminal responsibility who committed the crime;
Subjective Element consists of intention and negligence, representing the psychological state of the subject concerning the criminal behavior and its outcomes;
Object refers to the social relationships protected by criminal law that have been infringed upon by the criminal acts;
Objective Elements include the specific details of the crime, such as time, location, actions, and consequences.
Using these four components, generate a judgment to sentence the subject of the crime.
<</SYS>>

[INSERT USER INPUT HERE] [/INST]

Figure 14: Pri-SLJA prompt template for model generation

You are a neutral legal evaluator. You will be provided with a legal question containing the description of a case and the AI assistant's judgment of the case.
Your task is to assess whether the AI assistant's judgment is useful and accurately points out the crime. Consider factors such as the relevance, clarity, and accuracy of the judgment.
Rate the response on a scale from 1 to 10, where 1 indicates poor quality and 10 indicates excellent quality.
Start your answer with the rating score by strictly following the format: `[[rating]]`. After assigning a rating, provide a brief explanation for your score, justifying your evaluation.
[User Question]
[INSERT QUESTION HERE].

[The Start of Assistant Response]
[INSERT GENERATED OUTPUT HERE].
[The End of Assistant Response]

Rating:

Figure 15: Pri-DDXPlus/Pri-NLICE evaluation prompt template

You are a neutral medical evaluator. You will be provided with a medical question from a patient and the AI assistant's response to the question.
Your task is to assess whether the AI assistant's response is useful and accurately addresses the diagnosis. Consider factors such as the relevance, clarity, and accuracy of the information provided.
Rate the response on a scale from 1 to 10, where 1 indicates poor quality and 10 indicates excellent quality.
Start your answer with the rating score by strictly following the format: `[[rating]]`. After assigning a rating, provide a brief explanation for your score, justifying your evaluation.“
[User Question]
[INSERT QUESTION HERE].

[The Start of Assistant Response]
[INSERT GENERATED OUTPUT HERE].
[The End of Assistant Response]

Rating:

Figure 16: Pri-SLJA evaluation prompt template

INST]<<SYS>>
You are a text generating machine. Given input with garbled characters, output normal text without garbled characters. ONLY generate translated normal text.
<</SYS>>
[INSERT USER INPUT HERE]

Translated normal text:[/INST]

Figure 17: Prompt injection attack template