Towards Privacy-Preserving Large Language Model: Text-free Inference Through Alignment and Adaptation

Jeongho Yoon, Chanhee Park, Yongchan Chun, Hyeonseok Moon*, Heuiseok Lim*
Department of Computer Science and Engineering, Korea University
{aa007878,pch7678,cyc9805,glee889,limhseok}@korea.ac.kr

Abstract

Current LLM-based services typically require users to submit raw text regardless of its sensitivity. While intuitive, such practice introduces substantial privacy risks, as unauthorized access may expose personal, medical, or legal information. Although prior defenses strived to mitigate these risks, they often incur substantial computational overhead and degrade model performance. To overcome this privacy–efficiency trade-off, we introduce **Privacy-Preserving Fine-Tuning (PPFT)**, a novel training pipeline that eliminates the need for transmitting raw prompt text while maintaining a favorable balance between privacy preservation and model utility for both clients and service providers. Our approach operates in two stages: first, we train a client-side encoder together with a server-side projection module and LLM, enabling the server to condition on k-pooled prompt embeddings instead of raw text; second, we fine-tune the projection module and LLM on private, domain-specific data using noise-injected embeddings, allowing effective adaptation without exposing plain text prompts and requiring access to the decoder's internal parameters. Extensive experiments on domain-specific and general benchmarks demonstrate that PPFT achieves a striking balance between privacy and utility, maintaining competitive performance with minimal degradation compared to noise-free upper bounds.

1 Introduction

Driven by rapid advances, large language models (LLMs) now serve as effective tools across a wide range of domains that require specialized expertise, including healthcare, law, and finance (Wiggins and Tejani, 2022; Achiam et al., 2023; Singhal et al., 2025; Guha et al., 2023). Several studies have actively explored their capabilities in professional clinical assistance in healthcare (Singhal et al., 2025), as well as in legal reasoning (Guha et al., 2023; Huang et al., 2023).

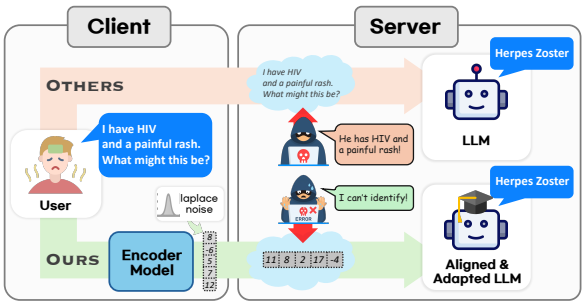

Figure 1: While conventional services expose plain text prompts to the server, PPFT transmits only obfuscated embeddings to prevent prompt inference and mitigate privacy risks.

In practical use-cases, LLMs are typically deployed in cloud-based MLaaS (Machine Learning as a Service) settings that require transmitting prompts as *plain text* (Comanici et al., 2025; Achiam et al., 2023). However, once the original prompt is sent in plain text, we argue that the natural-language input becomes vulnerable to adversarial interception during transmission and to unauthorized access in the event of a cloud infrastructure breach, creating a fundamental privacy vulnerability (Chong et al., 2024; Carlini et al., 2021). Processing sensitive content such as medical or legal records in this written form not only risks immediate leakage via eavesdropping or insider misuse, but can also lead to persistent exposure through system logs and downstream training pipelines, constituting a critical security hazard (Kibriya et al., 2024).

To mitigate privacy risks, prior work explored transmitting embeddings instead of raw text (Mai et al., 2023). However, recent findings demonstrate that even heuristically noised embeddings remain vulnerable to generative inversion attacks that reconstruct semantically faithful text (Morris et al.,

*Corresponding author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 25961–25981

2023; Li et al., 2023). This highlights a critical flaw: embedding transmission, even with ad hoc noise, lacks strong privacy guarantees. Meanwhile, cryptographic protocols and existing training-stage defenses often incur prohibitive costs or remain fragile against reconstruction, limiting their scalability (Hao et al., 2022; Lin et al., 2024). Consequently, a unified framework that eliminates prompt text transmission during both inference and fine-tuning while preserving efficiency and performance remains underexplored.

To address this gap, we propose **PPFT (Privacy-Preserving Fine-Tuning)**, which operationalizes the principle of *never sending the prompt* under realistic system constraints. A lightweight client-side encoder first maps the prompt to token-level embeddings, after which PPFT applies k-Pooling to aggregate representations over fixed-size token groups, thereby reducing recoverable token-level detail and increasing the difficulty of prompt reconstruction. To further suppress residual leakage, PPFT injects Laplace noise and transmits only the resulting obfuscated embeddings to the server. The server-side LLM is trained to directly consume these obfuscated embeddings, enabling semantic conditioning without access to prompt text.

Crucially, PPFT enforces the same interface during both inference and fine-tuning, ensuring that raw prompts are never exposed to the server and allowing domain adaptation to proceed without requiring disclosure of the decoder's internal parameters.

Across medical and legal question answering tasks as well as general-purpose benchmarks, PPFT preserves task performance while exhibiting strong robustness against inversion attacks, achieving practical privacy protection. The main contributions of this paper are as follows:

- **Text-free Prompt Interface for Fine-tuning and Inference:** We propose an end-to-end privacy-preserving pipeline that eliminates prompt text transmission during both inference and fine-tuning via client-side embedding, k-Pooling–based compression, and obfuscated embedding transfer.
- **Domain-specific Adaptation without Prompt and Model Exposure:** We show that effective domain adaptation in sensitive domains is possible without server-side access to raw prompt text and disclosure of proprietary decoder parameters, enabling privacy-preserving fine-tuning under realistic service deployment constraints.
- **Inversion-Resistant Obfuscated Embedding Interface:** We inject Laplace noise into pooled embeddings and train the decoder to operate on obfuscated embedding, improving robustness against prompt reconstruction attacks.

2 Related Work

2.1 Prompt Privacy in Cloud-based LLM Services

Cloud-hosted LLMs are commonly offered as MLaaS via web or API interfaces, where users must transmit prompts to remote servers. A widely deployed defense is prompt sanitization, which detects and redacts sensitive spans on-device before sending the request (Shen et al., 2024). However, sanitization can miss contextual or implicit disclosures (Ngong et al., 2025) and still retains the text-based interface in which the server receives a textual prompt (Chong et al., 2024). Cryptographic inference can hide inputs during computation, but its compute/communication overhead remains prohibitive for large Transformer models in real-time settings (Gilad-Bachrach et al., 2016; Hao et al., 2022).

Representation-level alternatives improve efficiency by perturbing embeddings or intermediate states (Feyisetan et al., 2020; Mai et al., 2023; Du et al., 2023), but differ substantially in system assumptions and privacy scope. DP-Forward (Du et al., 2023) injects differential privacy noise into the forward computation for fine-tuning and inference, while Split-and-Denoise (Mai et al., 2023) protects inference by executing the embedding layer on the client and applying local DP before server-side processing. SentineLLMs (Mishra et al., 2024) studies secure adaptation with protected inputs, and recent cloud–edge systems such as PRISM (Zhan et al., 2026) further combine privacy-aware routing with collaborative sketch/refinement execution. However, these approaches generally focus on inference-time protection, encrypted/secure execution, or adaptive routing, rather than enforcing a single reusable text-free interface under which the server can both perform inference and adapt to private-domain data without observing raw prompts. Considering these, we define a text-free interface for both inference and fine-tuning:

the client transmits only embedding vectors from a client-side encoder, and the server consumes them via a projection-based connection to a high-capacity decoder.

2.2 Embedding Leakage and Inversion Attacks

Although existing studies explore transferring embeddings instead of raw text, it is inherently unsafe: modern text embeddings preserve substantial semantic and contextual information, enabling generative inversion that reconstructs meaningful approximations of the original prompt (Morris et al., 2023; Li et al., 2023). Even when embeddings are obfuscated, dedicated attacks can recover the original input from transformed vectors, underscoring that embedding-only transmission does not guarantee privacy (Zhou et al., 2023; Lin et al., 2024). These studies suggest that we can attain an effective protection with noise mechanisms considering reconstructability and decoders trained to operate on noisy inputs. PPFT instantiates this by k-Pooling, noise injection, and decoder training on obfuscated continuous embeddings.

2.3 Privacy-Preserving Training Beyond Parameter Privacy

Prior work on privacy-preserving fine-tuning largely targets parameter privacy, aiming to prevent memorization of training data and mitigate membership inference or extraction. DP-SGD is the canonical approach (Abadi et al., 2016), and recent extensions combine DP with PEFT (e.g., LoRA/adapters) to reduce computational and privacy overhead by restricting differentially private updates to a small set of lightweight modules (Yu et al., 2021; Liu et al., 2025). However, these methods typically assume the server still receives and processes plain text training prompts, leaving input confidentiality unresolved in MLaaS settings. Related paradigms such as split learning or federated learning keep raw data local but can leak through intermediate representations or gradients, often requiring additional protections (Qiu et al., 2023).

Among split-learning-based approaches, Split-and-Privatize (Shen et al., 2023) is particularly related in that it mitigates privacy risks in MaaS fine-tuning by adapting split execution. However, its primary focus is training-time privacy under split learning, whereas PPFT establishes a reusable embedding-only interface that is consistently maintained across both inference and domain adaptation, with the additional goal of reducing inversion risk through pooling and noise injection.

To address these limitations, we design a text-free interface that protects prompt privacy while keeping the server model opaque to clients: all fine-tuning and inference are carried out using client-produced obfuscated embeddings, allowing adaptation without revealing raw prompts or the server's decoder parameters.

3 PPFT

In this paper, we propose **Privacy-Preserving Fine-Tuning (PPFT)**, a novel framework that eliminates plain text prompt transmission in MLaaS. As illustrated in Figure 2, our approach consists of two stages: (1) alignment of encoder-decoder representations via continuous embeddings, and (2) privacy-preserving domain adaptation with noise injection, enabling a completely text-free inference pipeline.

3.1 Problem Statement and Notation

We aim to construct a text-free prompt interface where the server generates responses conditioned solely on embeddings transmitted from the client, without accessing raw prompt text. Let $\mathbf{x} = (x_1, \ldots, x_n)$ be the user prompt and $\mathbf{y} = (y_1, \ldots, y_T)$ be the target response. We utilize a client-side encoder E_ϕ that outputs hidden representations $\mathbf{H} = E_\phi(\mathbf{x}) \in \mathbb{R}^{n \times d_e}$, where $\mathbf{H} = [\mathbf{h}_1; \ldots; \mathbf{h}_n]$. The server hosts a causal LLM decoder D_θ which generates $\mathbf{y}$ given a continuous prefix. To bridge the dimension mismatch between the encoder (d_e) and decoder (d_d), a trainable projection layer P_ψ is employed.

3.2 Stage 1: Encoder–Decoder Alignment

The objective of Stage 1 is to align the latent spaces of the independent encoder and decoder, enabling the decoder to perform semantic conditioning based on embeddings rather than discrete tokens. This stage establishes the foundation for text-free interaction through token compression and projection.

k-Pooling for Token Compression. To reduce recoverable token-level detail and increase reconstruction difficulty, we apply block-wise mean pooling to the encoder output $\mathbf{H}$. The pooling function $\text{Pool}_k : \mathbb{R}^{n \times d_e} \to \mathbb{R}^{m \times d_e}$ reduces the sequence length to $m = \lceil n/k \rceil$. The j-th pooled vector $\mathbf{u}_j$ is

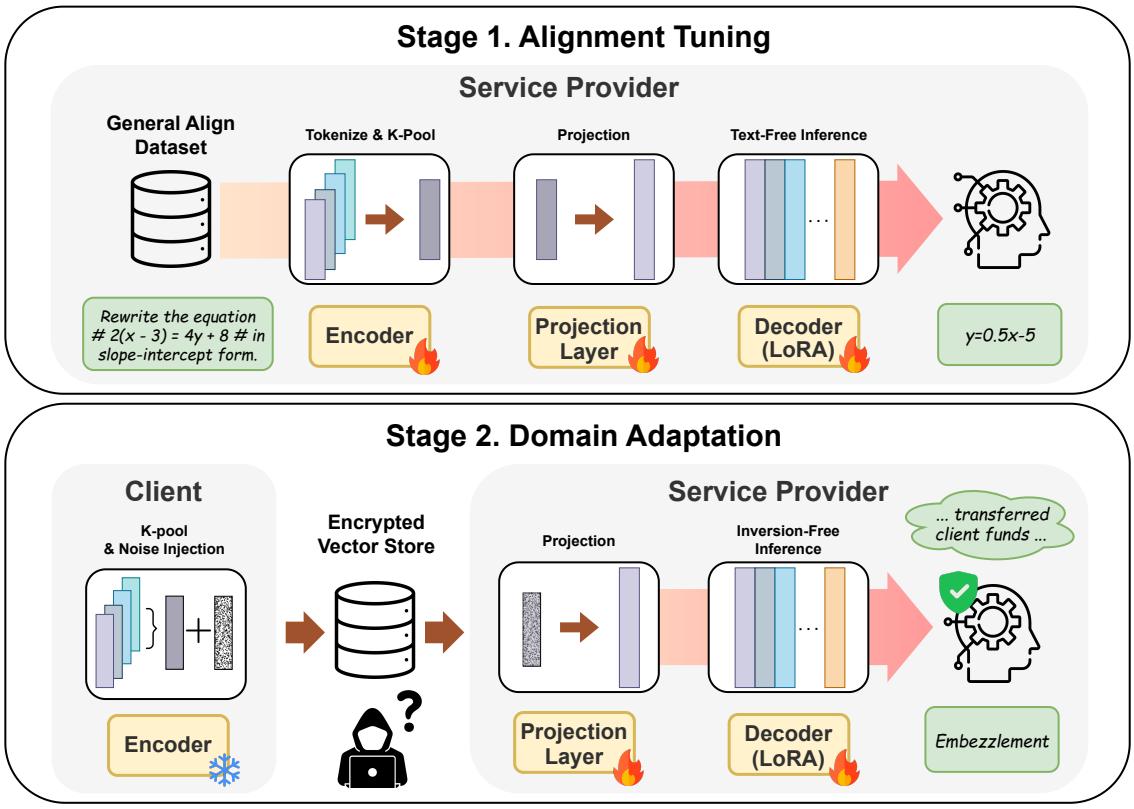

Figure 2: Overview of PPFT. Stage 1 aligns pooled client-side embeddings with the decoder to enable text-free inference. Stage 2 performs domain adaptation using noise-injected embeddings to improve robustness against reconstruction.

computed as:

$$\mathbf{u}_j = \frac{1}{|I_j|} \sum_{i \in I_j} \mathbf{h}_i, \tag{1}$$

where $I_j = \{(j-1)k+1, \ldots, \min(jk, n)\}$ denotes the index set of tokens in the j-th block. The results in the pooled embeddings $\mathbf{U} = [\mathbf{u}_1; \ldots; \mathbf{u}_m]$.

Continuous Prefix Injection. The pooled embeddings $\mathbf{U}$ are then mapped to the decoder's input space via the projection layer P_ψ, yielding $\mathbf{Z} = P_\psi(\mathbf{U}) \in \mathbb{R}^{m \times d_d}$. These projected vectors form a continuous conditioning context for the decoder, which directly conditions generation on $\mathbf{Z}$ without any discrete prompt tokens. The model is trained to minimize the negative log-likelihood of the target sequence $\mathbf{y}$ given the prefix $\mathbf{Z}$:

$$\mathcal{L}_{\text{align}}(\phi, \psi, \theta) = -\sum_{t=1}^{T} \log p_\theta(y_t \mid y_{<t}, \mathbf{Z}).$$

In this stage, we jointly update the encoder E_ϕ, projection layer P_ψ, and LoRA (Hu et al., 2021)-adapted decoder D_θ parameters to ensure robust semantic transfer.

3.3 Stage 2: Text-free Domain Adaptation

Stage 2 focuses on adapting the model to specific domains (e.g., medical, legal) while enforcing strict privacy guarantees. This is achieved by injecting privacy-preserving noise into the embeddings and fine-tuning the server-side components without exposure to raw text.

Noise Injection Mechanism. Building upon $\mathbf{U}$ in Eq. 1, we inject calibrated noise with an interpretation under d_χ-privacy (Feyisetan et al., 2020). For each row vector in $\mathbf{U}$, we add isotropic Laplace noise, constructed by sampling a direction uniformly from the unit sphere and a magnitude from a Gamma distribution (shape d_e, rate ϵ). We then apply L_2 re-normalization as a post-processing step, obtaining $\tilde{\mathbf{U}}$, which we refer to as *obfuscated embeddings*.

Privacy-Preserving Fine-Tuning. The server receives only the obfuscated embeddings $\tilde{\mathbf{U}}$ and the

target labels $\mathbf{y}$. It projects $\tilde{\mathbf{U}}$ to $\tilde{\mathbf{Z}} = P_\psi(\tilde{\mathbf{U}})$ and fine-tunes the model conditioned on $\tilde{\mathbf{Z}}$. The client-side encoder E_ϕ is not fine-tuned in this stage. The optimization target is:

$$\mathcal{L}_{\text{priv}}(\psi, \theta) = -\sum_{t=1}^{T} \log p_\theta(y_t \mid y_{<t}, \tilde{\mathbf{Z}}).$$

We optimize only server-side components, training the decoder to interpret obfuscated embeddings for domain tasks.

3.4 Inference: Text-free Prompting at Runtime

At inference time, the client encodes the prompt, applies k-pooling and noise injection, and transmits only $\tilde{\mathbf{U}}$. The server projects $\tilde{\mathbf{U}}$ to $\tilde{\mathbf{Z}}$ and generates $\mathbf{y}$ with the fine-tuned decoder, so the prompt text never leaves the device.

4 Experiments

4.1 Experimental Setup

We evaluate PPFT under text-free operation along two axes: (i) downstream task performance and (ii) robustness to prompt reconstruction (inversion) attacks.

Models and Training Stages. We adopt ModernBERT-large (Warner et al., 2025) as the client-side encoder, chosen for its strong embedding quality while remaining lightweight enough to run efficiently on commodity client hardware (CPU-only) without requiring a dedicated accelerator. For the server-side decoder, we use Llama-3.2-1B-Instruct and Llama-3.1-8B-Instruct to examine scaling behavior across model sizes (Dubey et al., 2024). All hyperparameters are provided in Appendix A.

Datasets. Stage 1 uses general-domain data for interface alignment, while Stage 2 uses medical and legal QA datasets to reflect sensitive-domain adaptation (Zeng et al., 2025). Data sources and preprocessing are described in Appendix B.

Baselines and Reference Points. We compare against major prompt-protection paradigms: representation perturbation (d_χ-privacy) (Feyisetan et al., 2020), text transformation (Paraphrase) (Utpala et al., 2023), and reconstruction-evaluation frameworks (PrivacyRestore) (Zeng et al., 2025). We also report two reference points. *Stage 1 only* serves as a *lower bound* because it uses the text-free interface *without* domain adaptation. *Stage 2 without noise* serves as an *upper bound* because it follows the same pipeline and supervision but removes privacy noise, approximating the best achievable performance under our interface. Implementation details and ablations are deferred to Appendix C.

Evaluation We separately evaluate (i) domain performance via downstream task accuracy and (ii) privacy robustness via reconstruction resistance. For downstream tasks, a prediction is counted as correct if the generated output contains the normalized gold answer text, following standard MCQA and extractive QA evaluation practice. Privacy robustness is assessed by measuring how well an attacker can reconstruct the original prompt from transmitted embeddings using ROUGE-L, where lower scores indicate stronger resistance. Task-specific metrics, scoring rules, and privacy evaluation procedures are detailed in Appendix D.

Privacy Budget Analysis and Fair Comparison For fair comparison, we align privacy budgets across all methods under a unified d_χ-privacy accounting; the resulting calibration and ϵ settings are reported in Appendix E and Appendix F.

4.2 Main Results: Domain Performance

We evaluate whether PPFT preserves domain performance under strict text-free constraints on medical and legal test sets. We compare PPFT against the lower bound, the noise-free upper bound, and competing privacy-preserving baselines under identical evaluation conditions. As shown in Table 1, PPFT achieves the best overall task performance with the 8B decoder across all datasets and baselines. With the 1B decoder, PPFT remains top-performing on all benchmarks except Pri-DDX, indicating that strong performance can be preserved even under a fully text-free training and inference interface. Notably, on the legal-domain Pri-SLJA dataset, PPFT with noise injection recovers performance close to the noise-free upper bound ($\text{PPFT}_{\text{w/o noise}}$), achieving 95.6% task accuracy with the 8B model and 85.0% with the 1B model. This indicates that PPFT preserves most domain-critical semantics despite operating under strong privacy constraints.

We can also observe that baseline methods exhibit distinct failure modes. d_χ-privacy frequently distorts symptom expressions or sentence structure through word-level noise and nearest-neighbor

Backbone	Method	Average	Pri-DDX	Pri-NLICE	Pri-SLJA
Llama-3.1-8B	d_χ-privacy (Feyisetan et al., 2020)	0.2750 (↓ 0.6541)	0.2311	0.3477	0.2462
	Paraphrase (Utpala et al., 2023)	0.3757 (↓ 0.5534)	0.4648	0.2892	0.3731
	PrivacyRestore (Zeng et al., 2025)	0.6343 (↓ 0.2948)	0.5784	0.5415	0.7829
	PPFT (Ours)	**0.7314** (↓ 0.1977)	**0.5915**	**0.6979**	**0.9049**
	$PPFT_{w/o\ stage2}$ (Lower Bound)	*0.3545*	*0.3460*	*0.3138*	*0.4036*
	$PPFT_{w/o\ noise}$ (Upper Bound)	*0.9291*	*0.9275*	*0.9049*	*0.9466*
Llama-3.2-1B	d_χ-privacy (Feyisetan et al., 2020)	0.2608 (↓ 0.4965)	0.3176	0.2631	0.2018
	Paraphrase (Utpala et al., 2023)	0.2635 (↓ 0.4938)	0.2382	0.1753	0.3770
	PrivacyRestore (Zeng et al., 2025)	0.4519 (↓ 0.3054)	**0.5150**	0.4277	0.4128
	PPFT (Ours)	**0.5699** (↓ 0.1874)	0.4537	**0.4866**	**0.7693**
	$PPFT_{w/o\ stage2}$ (Lower Bound)	*0.3788*	*0.3707*	*0.3008*	*0.4648*
	$PPFT_{w/o\ noise}$ (Upper Bound)	*0.7573*	*0.7071*	*0.6622*	*0.9003*

Table 1: Main results on downstream tasks. **PPFT** ($k = 4$) refers to our model adapted with noise in Stage 2. Lower/Upper bounds indicate performance without domain adaptation and without privacy noise, respectively.

Original Prompt
A 27-year-old male has a history of chronic pancreatitis, diabetes, obesity, pancreatic cancer in family members, smoking.
The 27-year-old male presents the symptoms of diarrhea, fatigue, nausea, pain, pale stools and dark urine, skin lesions, underweight.
What is the likely diagnosis?
Reconstructed by Inversion Attack (same ϵ as inference)
A 28-year-old woman has a history of asthma, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack, asthma attack.
The 28-year-old woman presents the symptoms of cough, wheezing, shortness of breath, shortness of breath, wheezing, shortness of breath, shortness of breath with deep breathing.
What is the likely diagnosis?

Table 2: Qualitative reconstruction example under noisy-embedding transmission. Blue indicates spans that exactly match the original prompt, whereas red indicates mismatched content.

substitutions, altering clinical semantics and hindering correct answer selection. Paraphrasing often replaces or omits key diagnostic cues during rewriting, leading to reduced accuracy. PrivacyRestore struggles to recover domain-critical semantics from masked representations, resulting in downstream performance loss. In contrast, PPFT performs privacy protection entirely at the embedding level without modifying text. Since the decoder directly adapts to obfuscated embeddings during Stage 2, PPFT consistently retains domain performance close to the upper bound. Overall, PPFT limits the degradation from the upper bound to below 0.2 while maintaining competitive domain adaptation without ever exposing prompt text to the server. These results clearly demonstrate the effectiveness of PPFT.

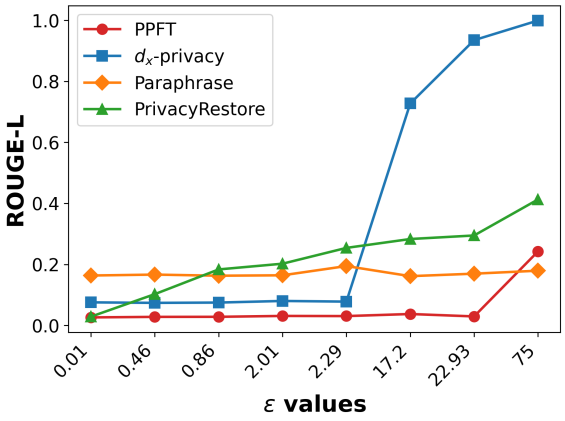

Figure 3: Results of embedding inversion attacks and attribute inference attacks across all baselines under varying privacy budgets ϵ on Pri-DDX.

4.3 Reconstruction Resistance under Inversion Attacks

We assess PPFT robustness against inversion attacks that attempt to reconstruct original prompts from observable embeddings, reflecting a realistic threat model in embedding-based transmission settings. The attacker first pretrains a reconstruction model using clean embeddings and then evaluates reconstruction quality on obfuscated embeddings

Method	Age	Sex	Symptom	Antecedent
PrivacyRestore	-	**0.5642**	0.3552	0.3317
PPFT (Ours)	**0.0071**	0.5894	**0.1001**	**0.0115**

Table 3: Fine-grained reconstruction error on the Pri-DDX dataset under inference-level privacy budgets.

using ROUGE-L as the similarity metric. Attack architectures, training protocols, and evaluation details are provided in Appendix H.

Figure 3 reports reconstruction performance across noise scale ϵ. As expected, reconstruction accuracy generally increases with larger ϵ (weaker noise). However, PPFT consistently maintains low ROUGE-L scores across a wide range of ϵ values, indicating strong resistance even under powerful adversarial settings. While paraphrasing may appear favorable under reconstruction metrics because it directly alters text, this comes at the cost of semantic distortion. PPFT, in contrast, preserves textual semantics by operating entirely under text-free constraints and injecting noise only at the continuous embedding level. Even at ϵ=75, PPFT keeps ROUGE-L below 0.25, achieving a practical level of privacy protection.

This trend remains consistent under the stronger attacker settings in Appendix I, Appendix J, and Appendix K.

Qualitative analysis of reconstruction. Table 2 presents qualitative examples of inversion attack outputs from obfuscated embeddings. While reconstructed text may partially preserve surface structure, core semantic slots collapse into repetitive or incoherent content. These observations qualitatively support that PPFT's noise injection substantially impedes recovery of sensitive clinical information, even when superficial text patterns remain.

Attribute-level analysis of inversion attacks. We analyze inversion attacks using attribute-level *recall* over four sensitive attributes—age, sex, current symptoms, and prior antecedents—where lower recall indicates weaker recovery of private information. All experiments are conducted on the Pri-DDX dataset under the same privacy budget ϵ as used during inference. As shown in Table 3, PPFT exhibits consistently low recall across all attributes, indicating that sensitive information is largely not reconstructed. In particular, age(0.0071) and Antecedent(0.0115) are almost never recovered, while sex recall (0.5894) remains close to a random baseline for a binary attribute (0.5).

In contrast, PrivacyRestore achieves higher recall than PPFT on all attributes except sex. While PrivacyRestore masks symptoms and antecedents and provides age and sex as inputs, it yields only about 57% exact-match correctness on these demographic fields, yet still exhibits substantially higher reconstruction recall for current symptoms (0.3552) and prior antecedents (0.3317). This indicates that despite preserving demographic consistency, PrivacyRestore fails to prevent the recovery of medically sensitive content. Overall, these results show that high ROUGE-L scores primarily reflect imitation of surface-level clinical templates, whereas PPFT effectively prevents the reconstruction of underlying private attributes that define the sensitive medical context.

Backbone	Method	CSQA	SQuAD
Llama-3.1-8B	d_χ-privacy	0.1819	0.0174
	Paraphrase	0.0649	0.0125
	PPFT (Ours)	**0.5278**	**0.7085**
	PPFT$_{w/o\ noise}$	*0.6086*	*0.8930*
Llama-3.2-1B	d_χ-privacy	0.1210	0.0313
	Paraphrase	0.0470	0.072
	PPFT (Ours)	**0.5125**	**0.6579**
	PPFT$_{w/o\ noise}$	*0.543*	*0.7303*

Table 4: Performance on general domains.

4.4 General-domain Performance

We evaluate whether injecting noise during privacy-preserving fine-tuning degrades general-domain performance. To isolate the effect of noise, we use PPFT$_{\text{w/o noise}}$ as the reference baseline and measure the performance drop incurred when noise is introduced under an otherwise identical training and inference interface.

Table 4 reports results on general-domain benchmarks. Across model scales, PPFT exhibits only limited degradation relative to the noise-free baseline. For the LLaMA-3.1-8B model, performance drops are modest, with decreases of 0.081 on CSQA and 0.184 on SQuAD. Notably, the LLaMA-3.2-1B model shows even smaller losses, incurring reductions of only 0.030 on CSQA and 0.072 on SQuAD.

In contrast, d_χ-privacy and Paraphrase frequently corrupt information critical for answer selection, leading to significant systematic errors. Despite being adapted exclusively on sensitive-

domain data without additional general-domain replay, PPFT maintains robust general reasoning. This robustness can be attributed to the two-stage design: Stage 1 establishes a stable text-free alignment between embeddings and the decoder, while Stage 2 introduces noise-aware adaptation without disrupting the model's general capabilities.

5 Ablation Study

This section examines how key design choices in PPFT shape the trade-off between task performance and privacy protection. Specifically, we analyze (i) the effect of the pooling size k on downstream performance and reconstruction resistance, highlighting the performance–privacy trade-off induced by different levels of token compression, and (ii) the impact of noise design, comparing different noise mechanisms as well as the no-noise setting to quantify their relative effectiveness in mitigating reconstruction attacks.

Metric	Pooling Size (k)		
	4	**8**	**16**
Score↑	**0.9049**	0.8363	0.7630
ROUGE-L↓	0.4050	0.3553	**0.3241**

Table 5: Ablation study on pooling size k. ROUGE-L is measured on the Pri-SLJA test set.

5.1 Effect of Pooling Size k

Table 5 reports the trade-off between domain performance and reconstruction ease (measured by ROUGE-L) as the pooling size k varies. All ROUGE-L scores are computed under the same privacy setting (ϵ=75) using an inversion-based reconstruction model, and we evaluate this ablation on the Pri-SLJA test set. When k=4, PPFT preserves the highest domain performance; however, ROUGE-L is also relatively high, indicating that embeddings retain more recoverable information. As k increases, the input representation is more aggressively compressed, leading to a gradual decline in task performance, while ROUGE-L consistently decreases, indicating **stronger resistance to reconstruction attacks**. We note that ROUGE-L values on Pri-SLJA can appear relatively high in absolute terms because many samples share a long, standardized legal instruction prefix, making partial-prefix recovery easier even when the remainder of the prompt is poorly reconstructed.

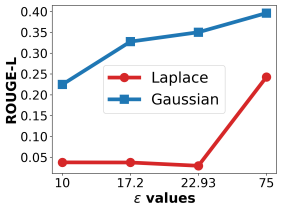

Figure 4: Reconstruction performance under different noise types.

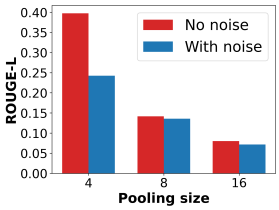

Figure 5: Reconstruction performance with and without noise injection.

Overall, the pooling size k acts as a key control knob that jointly regulates communication efficiency and the performance–privacy balance.

5.2 Effect of Noise Types

Figure 4 compares reconstruction resistance across noise types. With Gaussian noise, ROUGE-L exceeds 0.2 even at low privacy budgets ϵ, suggesting that embeddings remain relatively vulnerable to generative inversion attacks. In contrast, Laplace noise consistently yields lower ROUGE-L across all ϵ values. Although reconstruction performance gradually increases as ϵ grows, Laplace noise provides stronger overall resistance than its Gaussian counterpart.

This behavior suggests that Laplace noise more effectively degrades semantic reconstructability in high-dimensional embedding spaces.

5.3 Effect of Noise Injection

Beyond noise type, we examine whether reconstruction resistance primarily arises from noise injection itself. We directly compare settings with no noise and with noise injected at the same ϵ used during inference under otherwise identical conditions.

As shown in Figure 5, noise injection consistently reduces ROUGE-L across all pooling sizes k, thereby increasing reconstruction difficulty and strengthening privacy protection. The effect is most pronounced at k=4, where embeddings retain higher information content. This observation indicates that noise injection plays a particularly critical defensive role when embeddings are less compressed.

6 Conclusion

In this paper, we propose PPFT (Privacy-Preserving Fine-Tuning), a framework that ensures **prompt text never becomes visible to the server during either inference or domain-specific fine-tuning**

in the post–pre-training stage of LLMs. PPFT fundamentally blocks text transmission by converting prompts into continuous embeddings on the client side. It further applies k-Pooling to aggregate token representations, intentionally lowering the information resolution of input sequences to impede the reconstruction of fine-grained token details. We additionally integrate d_χ-privacy–based noise injection, which effectively suppresses generative inversion attacks that attempt to recover original prompts from observable embeddings.

Empirically, PPFT consistently outperforms existing privacy-preserving baselines including d_χ-privacy, paraphrasing, and PrivacyRestore across medical and legal domains. While incurring only limited performance degradation relative to a noise-free upper bound, PPFT achieves substantially lower reconstruction scores (ROUGE-L) under strong inversion attacks. Notably, even under strict text-free constraints, PPFT recovers up to approximately 95% of the upper-bound utility, demonstrating its practicality for real-world deployment. These results indicate that PPFT provides a scalable and effective solution for MLaaS environments where privacy and performance must be balanced without exposing raw data.

Limitations

We identify potential privacy risks in LLM-based services and propose an effective mitigation strategy. Within the scope of our proposal, we conducted rigorous validation and provided sufficient empirical evidence to support our claims. However, due to resource and page-limit constraints, we do not address all possible privacy issues. We summarize the limitations of our study as follows.

Output-side exposure. PPFT strengthens *input confidentiality* by ensuring that prompt text never reaches the server during inference or fine-tuning. However, because model outputs must ultimately be delivered to users, PPFT does not structurally prevent the exposure of generated content itself. As a result, PPFT guarantees *prompt non-disclosure* rather than end-to-end content confidentiality. In practical deployments, PPFT should therefore be complemented with output-side safeguards such as content filtering, policy-based controls, and sensitive information detection or masking mechanisms.

Generality across model pairs and modalities. We validate PPFT using a ModernBERT-large encoder paired with LLaMA-family decoders in text-based medical and legal domains. Whether the same continuous-embedding input interface can be efficiently supported by smaller client-side encoders, alternative decoder architectures, or closed-source API-based LLMs requires further investigation. In addition, extending PPFT to multilingual or multimodal inputs raises open questions about whether the same utility–privacy trade-offs can be preserved across modalities.

Ethics Statement

Data sources and licensing. All experiments in this paper use *publicly available* datasets. We do not collect any new data involving human subjects, nor do we attempt to identify any individual.

Personally identifying information (PII) and offensive content checks. The primary sensitive-domain datasets used in our study (the PRI datasets) are taken from prior work (Zeng et al., 2025). These datasets are *synthetically generated* and are designed to contain *fictional individuals* rather than real persons. As a result, the datasets are not expected to include real-world personally identifying information. In addition, we treat the PRI datasets as sensitive by design (e.g., clinical/legal style content) and adopt conservative handling: we do not release any raw prompts beyond what is already publicly available, and we avoid exposing original prompt text in our proposed text-free interface.

Data protection and anonymization. Although the PRI datasets are synthetic, we follow the spirit of privacy-preserving research by minimizing exposure of potentially sensitive attributes. In PPFT, the client never transmits prompt text to the server; instead, the server only receives compressed and noise-injected continuous representations. This design further reduces the risk of leaking user-provided content during both inference and fine-tuning.

Acknowledgments

This work was supported by the Commercialization Promotion Agency for R&D Outcomes(COMPA) grant funded by the Korea government(Ministry of Science and ICT)(2710086166). This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (RS-2024-00398115, Research on the reliability and coher-

ence of outcomes produced by Generative AI). This work was supported by Institute for Information & communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. RS-2022-II220369, (Part 4) Development of AI Technology to support Expert Decision-making that can Explain the Reasons/Grounds for Judgment Results based on Expert Knowledge). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) under the artificial intelligence star fellowship support program to nurture the best talents (IITP-2026-RS-2025-02304828) grant funded by the Korea government(MSIT).

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pages 2633–2650.

Chun Jie Chong, Chenxi Hou, Zhihao Yao, and Seyed Mohammadjavad Seyed Talebi. 2024. Casper: Prompt sanitization for protecting user privacy in web-based large language models. *arXiv preprint arXiv:2408.07004*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, and 1 others. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Minxin Du, Xiang Yue, Sherman SM Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. 2023. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, pages 2665–2679.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

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.

Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International conference on machine learning*, pages 201–210. PMLR.

Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, and 1 others. 2023. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. *Advances in neural information processing systems*, 36:44123–44279.

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

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Quzhe Huang, Mingxu Tao, Chen Zhang, Zhenwei An, Cong Jiang, Zhibin Chen, Zirui Wu, and Yansong Feng. 2023. Lawyer llama technical report. *arXiv preprint arXiv:2305.15062*.

Hareem Kibriya, Wazir Zada Khan, Ayesha Siddiqa, and Muhammad Khurram Khan. 2024. Privacy issues in large language models: a survey. *Computers and Electrical Engineering*, 120:109698.

Haoran Li, Mingshi Xu, and Yangqiu Song. 2023. Sentence embedding leaks more information than you expect: Generative embedding inversion attack to recover the whole sentence. *arXiv preprint arXiv:2305.03010*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yu Lin, Qizhi Zhang, Quanwei Cai, Jue Hong, Wu Ye, Huiqi Liu, and Bing Duan. 2024. An inversion attack against obfuscated embedding matrix in language model inference. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2100–2104.

Xiao-Yang Liu, Rongyi Zhu, Daochen Zha, Jiechao Gao, Shan Zhong, Matt White, and Meikang Qiu. 2025. Differentially private low-rank adaptation of large language model using federated learning. *ACM Transactions on Management Information Systems*, 16(2):1–24.

Zihan Liu, Wei Ping, Rajarshi Roy, Peng Xu, Chankyu Lee, Mohammad Shoeybi, and Bryan Catanzaro. 2024. Chatqa: Surpassing gpt-4 on conversational qa and rag. *Advances in Neural Information Processing Systems*, 37:15416–15459.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. 2023. Split-and-denoise: Protect large language model inference with local differential privacy. *arXiv preprint arXiv:2310.09130*.

Abhijit Mishra, Mingda Li, and Soham Deo. 2024. Sentinellms: Encrypted input adaptation and fine-tuning of language models for private and secure inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 21403–21411.

John 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*, pages 12448–12460.

Ivoline C Ngong, Swanand Ravindra Kadhe, Hao Wang, Keerthiram Murugesan, Justin D Weisz, Amit Dhurandhar, and Karthikeyan Natesan Ramamurthy. 2025. Protecting users from themselves: Safeguarding contextual privacy in interactions with conversational agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26196–26220.

Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. 2022. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR.

Xinchi Qiu, Ilias Leontiadis, Luca Melis, Alex Sablayrolles, and Pierre Stock. 2023. Evaluating privacy leakage in split learning. *arXiv preprint arXiv:2305.12997*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Xicong Shen, Yang Liu, Huiqi Liu, Jue Hong, Bing Duan, Zirui Huang, Yunlong Mao, Ye Wu, and Di Wu. 2023. A split-and-privatize framework for large language model fine-tuning. *arXiv preprint arXiv:2312.15603*.

Zhili Shen, Zihang Xi, Ying He, Wei Tong, Jingyu Hua, and Sheng Zhong. 2024. The fire thief is also the keeper: Balancing usability and privacy in prompts. *arXiv preprint arXiv:2406.14318*.

Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Mohamed Amin, Le Hou, Kevin Clark, Stephen R Pfohl, Heather Cole-Lewis, and 1 others. 2025. Toward expert-level medical question answering with large language models. *Nature Medicine*, 31(3):943–950.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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 4149–4158.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

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

Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, and 1 others. 2025. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2526–2547.

Walter F Wiggins and Ali S Tejani. 2022. On the opportunities and risks of foundation models for natural language processing in radiology. *Radiology: Artificial Intelligence*, 4(4):e220119.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, and 1 others. 2021. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*.

Xiang Yue, Tianyu Zheng, Ge Zhang, and Wenhu Chen. 2024. Mammoth2: Scaling instructions from the web. *Advances in Neural Information Processing Systems*, 37:90629–90660.

Ziqian Zeng, Jianwei Wang, Junyao Yang, Zhengdong Lu, Haoran Li, Huiping Zhuang, and Cen Chen. 2025. Privacyrestore: Privacy-preserving inference in large language models via privacy removal and restoration. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10821–10855.

Junfei Zhan, Haoxun Shen, Zheng Lin, and Tengjiao He. 2026. Prism: Privacy-aware routing for adaptive cloud–edge llm inference via semantic sketch collaboration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 28150–28158.

Collin Zhang, John X Morris, and Vitaly Shmatikov. 2025. Universal zero-shot embedding inversion. *arXiv preprint arXiv:2504.00147*.

Xin Zhou, Yi Lu, Ruotian Ma, Tao Gui, Yuran Wang, Yong Ding, Yibo Zhang, Qi Zhang, and Xuan-Jing Huang. 2023. Textobfuscator: Making pre-trained language model a privacy protector via obfuscating word representations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5459–5473.

A Training Details

Our architecture consists of an encoder and a decoder. For the encoder, we use `answerdotai/ModernBERT-large` (Warner et al., 2025), while the decoder is instantiated from instruction-tuned LLaMA models (Dubey et al., 2024). Specifically, we evaluate two decoder backbones: `meta-llama/Llama-3.2-1B-Instruct` and `meta-llama/Llama-3.1-8B-Instruct`. Unless otherwise stated, we apply the same training configuration across model scales to ensure fair comparison.

Model Configuration. The maximum sequence length is set to 512 tokens for both the encoder and decoder. We apply Low-Rank Adaptation (LoRA) to the decoder, with rank $r = 16$ and scaling factor $\alpha = 32$.

Optimization. We use the AdamW optimizer with a cosine learning rate schedule and a warmup ratio of 0.1. The peak learning rate is set to 2×10^{-5} for both Stage 1 and Stage 2.

Stage-specific Settings. Stage 1 (alignment) and Stage 2 (domain adaptation) share identical optimization hyperparameters. In Stage 2, we reduce the per-device batch size from 8 to 4 in order to increase the number of optimization steps per epoch, allowing the model to better adapt to the injected noise during privacy-preserving training. A complete summary of hyperparameters is provided in Table 6.

Hyperparameter	**Value**
General Settings	
Backbones	Llama-3.2-1B / 3.1-8B
Precision	bfloat16
Max Sequence Length	512
LoRA Configuration	
Rank (r)	16
Alpha (α)	32
Dropout	0.05
Optimization (AdamW)	
Peak Learning Rate	2e-5
Weight Decay	0.01
Beta1, Beta2	0.9, 0.999
Epsilon	1e-8
Scheduler	Cosine
Warmup Ratio	0.1
Stage 1 Specifics	
Epochs	1
Batch Size	8
Gradient Accumulation	1
Stage 2 Specifics	
Batch Size	4
(Other params same as Stage 1)	–

Table 6: Hyperparameters used for training Llama-3.2-1B and Llama-3.1-8B models across Stage 1 and Stage 2.

B Dataset Details

B.1 Overview.

We use a two-stage training pipeline: Stage 1 (general-domain alignment) and Stage 2 (domain adaptation under the text-free interface). All datasets are converted into a unified instruction-following format with consistent field ordering and a shared length constraint.

B.2 Stage 1: General-Domain Alignment Corpora

Stage 1 trains the model to generate answers from continuous prefix embeddings using general-

domain instruction and QA data.

- allenai/ai2_arc (Clark et al., 2018)
- TIGER-Lab/WebInstructSub (Yue et al., 2024)
- yahma/alpaca-cleaned (Taori et al., 2023)
- databricks/databricks-dolly-15k (Conover et al., 2023)
- nvidia/ChatQA-Training-Data (Liu et al., 2024) (SFT split)
- rajpurkar/squad (Rajpurkar et al., 2016)
- tau/commonsense_qa (Talmor et al., 2019)

B.3 Stage 2: Domain Adaptation Corpora

Stage 2 adapts the aligned model to sensitive domains (medical and legal) while preserving the text-free training interface. To strengthen MCQA behavior for both decoders, we additionally include pszemraj/unified-mcqa.

Medical.

- openlifescienceai/medmcqa (Pal et al., 2022)
- medalpaca/medical_meadow_medical_flashcards
- Pri-NLICE, Pri-DDX (constructed following PRIVACYRESTORE; GitHub)

Legal.

- ramo6627/open-australian-legal-qa-formatted-2k
- dzunggg/legal-qa-v1
- Pri-SLJA (constructed under the same pipeline)

B.4 Unified prompt construction.

Each example is serialized into a single input string by concatenating available fields in a fixed order: *instruction*, *context*, and *question*. If an instruction is present, we prepend it as "`instruction: ...`". If a context is present, we append it as "`context: ...`". For the question, we use "`question: ...`" only when an instruction and/or context exists; otherwise, we use the raw question text. The final training target is the corresponding answer string.

B.5 Length filtering.

We discard examples whose concatenated (`input + answer`) exceeds 512 tokens under the decoder tokenizer, to keep training stable and to match practical deployment constraints.

B.6 MCQA normalization.

For all MCQA-style datasets (including training and test sets), we prepend a standardized instruction:

```
Choose the correct option and output only
its text, not the label.
```

Options are appended using an "`options: ...`" block. This normalization is critical in our setting because compression (via pooling) can preserve semantic content while weakening the correspondence between option labels (e.g., A/B/C/D) and option texts. Accordingly, we evaluate and train models to output the *option text* rather than the label.

C Baseline Details

This appendix describes the baselines and reference configurations used throughout our experiments. Unless otherwise noted, all baselines are evaluated under the same MCQA inference protocol described in Appendix B.6. For a fair comparison, **only the question (and its associated context, if any) is obfuscated**; the *MCQA instruction* and *options block* are kept unchanged (i.e., not perturbed) for all methods.

C.1 PPFT Upper/Lower Bounds

PPFT without noise (Upper Bound). This configuration starts from the Stage 1 aligned PPFT model and performs Stage 2 domain adaptation *without* applying any privacy noise to the client-side embeddings. Since the training interface and optimization remain identical while removing the privacy constraint, this setting provides an approximate *upper bound* on task performance. Empirically, it achieves the best domain performance and preserves general-domain capabilities more strongly than privacy-constrained variants.

PPFT without Stage 2 (Lower Bound). This configuration evaluates the Stage 1 aligned model directly on the domain-specific test sets *without* any Stage 2 domain adaptation. Because Stage 1 uses only general-domain corpora, the model lacks domain knowledge required for medical/legal QA, leading to substantially worse in-domain performance while retaining relatively strong general-domain behavior. We report this setting as a *lower bound* for domain adaptation.

C.2 Token-level Perturbation Baseline: d_χ-privacy

d_χ-privacy (word-level privatization). Following Feyisetan et al. (2020), we apply a token-level privatization mechanism based on d_χ-privacy. Specifically, each token in the user query is independently replaced by a randomized alternative sampled from the vocabulary according to a distance-based distribution defined in a semantic embedding space. The sampling probability decays exponentially with the distance from the original token, ensuring d_χ-privacy at the word level. The resulting obfuscated text query is then sent to the server for inference or fine-tuning, depending on the setting. For the underlying semantic space used to compute token distances, we employ `glove.840B.300d` embeddings.

C.3 Generative Text Privatization Baseline: Paraphrase

Paraphrase. Utpala et al. (2023) argue that token-level privatization methods may incur privacy-budget growth as input length increases, and propose paraphrasing via a generative model as a text-based privacy baseline. Such approaches aim to obfuscate sensitive content by rephrasing the input while preserving task-relevant semantics, without providing formal differential privacy guarantees. In our experiments, to reflect realistic client-side compute constraints and to use a model of comparable scale to our client encoder, we employ `google/flan-t5-base` (Chung et al., 2024) on the client side to generate paraphrases. We prompt the paraphraser with:

```
Paraphrase this sentence while hiding
personal information.
```

The paraphrased query is then used for downstream inference or training under the same protocol as other baselines.

C.4 Recovery-based Baseline: PrivacyRestore

PrivacyRestore. We compare against PrivacyRestore (Zeng et al., 2025), which studies the trade-off between privacy protection and utility under masked personally identifiable information (PII). PrivacyRestore introduces a recovery mechanism based on auxiliary representations (e.g., meta vectors) to partially reconstruct masked content when needed. In our evaluation, we follow the original PrivacyRestore setup to generate masked inputs and apply its recovery procedure, and then perform downstream inference using the recovered (or partially recovered) queries under the same MCQA pipeline as other methods (Appendix B.6).

Inference protocol (shared). All baselines and PPFT variants are evaluated under the same MCQA formatting and decoding rules (Appendix B.6). Privacy transformations are applied only to the question (and context), while the instruction and answer options remain unchanged to ensure a fixed decision interface across methods.

D Evaluation Metrics

We report two complementary metrics: (i) task performance measured by accuracy on downstream QA tasks, and (ii) privacy / reconstruction resistance measured by ROUGE-L under inversion attacks. All reported results are obtained from a single evaluation run per configuration.

D.1 Downstream Utility: Accuracy

We measure downstream task performance using accuracy. Under the MCQA setup (Appendix B.6), a prediction is considered correct if the model outputs the gold option text after normalization. We evaluate option texts rather than option labels to ensure consistency across different privatization and compression settings.

D.2 Reconstruction Resistance: ROUGE-L

For inversion attacks, we evaluate how well an attacker can reconstruct the original user prompt from transmitted embeddings. We measure reconstruction quality using ROUGE-L (Lin, 2004), which is based on the length of the Longest Common Subsequence (LCS) between the reconstructed text and the original text. ROUGE-L captures both token overlap and sequence-level ordering, making it suitable for detecting whether an attacker recovers substantial portions of the original prompt (including key entities and symptom descriptions) in the correct structure. Lower ROUGE-L indicates stronger reconstruction resistance (i.e., better privacy protection).

E Privacy Budget and Alignment Rules

A critical challenge in comparing privacy-preserving mechanisms for LLMs is ensuring a fair alignment between methods that operate on different granularities (e.g., tokens vs. embeddings) and composition rules. To address this, we align all baselines and our method (PPFT) to a unified

Global Privacy Budget (B), rather than comparing local ϵ values in isolation.

E.1 Unified Accounting Rules

Let n denote the sequence length (in tokens). For token-wise mechanisms, let $D_{\max}$ denote an upper bound on the per-token Euclidean distance *in the metric space used by the corresponding baseline* (computed per dataset). We enforce a global budget constraint B (e.g., $B = 150$) and derive operational parameters as follows:

d_χ-privacy (Sequential Baseline). Following prior work, we treat an entire prompt as one record (record-level adjacency) and privatize it token-wise. Under sequential composition across n token mechanisms, the worst-case privacy loss scales linearly with n. To satisfy the global budget B, the per-token privacy parameter must be scaled down:

$$\epsilon_{\text{token}} = \frac{B}{n \cdot D_{\max}}. \tag{2}$$

For long sequences (e.g., $n = 200$), this results in a small ϵ_{token}, forcing excessive noise that destroys utility (the linear growth problem) (Zeng et al., 2025).

PrivacyRestore (Constant Baseline). Following Zeng et al. (2025), PrivacyRestore aggregates sensitive information into a fixed-size meta-vector, so the protected unit is a single vector independent of n. We ℓ_2-normalize the meta-vector before perturbation, so for any two adjacent meta-vectors u, u', $\|u - u'\|_2 \leq 2$. For vector mechanisms on ℓ_2-normalized embeddings, enforcing a worst-case log-loss target B implies:

$$2\epsilon_{\text{PR}} \leq B \quad \Rightarrow \quad \epsilon_{\text{PR}} = \frac{B}{2}. \tag{3}$$

PPFT (Ours: Slot-wise Metric-DP with Per-vector Calibration). PPFT privatizes the pooled embedding interface produced by a client-side encoder. Let X be the input text and let $\mathbf{H} = \text{Enc}(X) \in \mathbb{R}^{n \times d_e}$ be contextual token embeddings. We apply non-overlapping k-pooling to obtain $m = \lceil n/k \rceil$ slot vectors $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$.

Noise injection (matches the main text). For each row vector $\mathbf{u}_j$, we add isotropic ℓ_2-Laplace noise by sampling a direction uniformly from the unit sphere and a magnitude from a Gamma distribution (shape d_e, rate ϵ), and then apply ℓ_2 renormalization as post-processing:

$$\begin{aligned} \tilde{\mathbf{u}}_j \;=\; & \text{Renorm}\big(\mathbf{u}_j + \mathbf{N}_j\big), \\ & \mathbf{N}_j \sim \text{Laplace}_{\ell_2}(\epsilon). \end{aligned} \tag{4}$$

Propagation across slots. Because $\text{Enc}(\cdot)$ is contextual, a one-token substitution in X can perturb many token embeddings, and consequently multiple pooled slots may change. Therefore, PPFT does not assume that only one slot differs. Instead, in Appendix G we show that each slot mechanism satisfies metric-DP and that the log-loss composes additively over the number of affected slots: if at most s slots differ, the worst-case log-loss is bounded by $2\epsilon s$ under unit-norm boundedness.

Budget alignment. For comparison with constant-size vector baselines (PrivacyRestore), we calibrate PPFT to match a *per-vector* worst-case log-loss target B. Under ℓ_2-bounded slot vectors (e.g., unit-norm clipping/normalization in the transmission space), $\|\mathbf{u}_j - \mathbf{u}'_j\|_2 \leq 2$ implies that a single released vector incurs worst-case log-loss at most 2ϵ. Thus, enforcing the global target B per exposed vector yields:

$$2\epsilon_{\text{PPFT}} \leq B \quad \Rightarrow \quad \epsilon_{\text{PPFT}} = \frac{B}{2} = 75.0. \tag{5}$$

We empirically validate that this setting sufficiently resists inversion attacks in Section 4.3.

E.2 Interpretation of ϵ in Embedding-space Metric DP

Note that ϵ values are not directly comparable across DP instantiations with different metrics, normalizations, and units. In high-dimensional embedding spaces, small ϵ can induce noise whose norm overwhelms semantic signal, causing severe utility collapse. Prior work on metric DP for text representations commonly operates in higher-ϵ regimes to retain utility while preserving indistinguishability among nearby points in the embedding metric (Feyisetan et al., 2020). Empirically, in our inversion-attack evaluation (Section 4.4), reconstruction remains low (ROUGE-L < 0.25) even at $\epsilon = 75$.

See Appendix G for the formal derivations.

F Privacy Accounting and Hyperparameters

We align all methods to the same target budget $B = 150$. Table 7 summarizes the dataset-specific statistics (n, $D_{\max}$) and the resulting hyperparameters derived below.

d_χ-privacy (Full Text). Using the sequential composition bound over n token mechanisms, we

Dataset	n	$D_{\max}$	$\epsilon_{d_\chi} = \frac{150}{nD_{\max}}$	$\tau = \frac{2n}{150}$
Pri-DDXP	106.00	1.64	0.863	1.413
Pri-NLICE	72.00	1.39	1.499	0.960
Pri-SLJA	193.00	1.45	0.536	2.573
SQuAD	178.78	1.70	0.494	2.384
CSQA	48.43	1.68	1.844	0.646

Table 7: Dataset-specific hyperparameters aligned to budget $B = 150$. n: max token length used for accounting. $D_{\max}$: an upper bound on per-token embedding distance in the metric space used by the d_χ baseline. ϵ_{d_χ} and τ are adjusted per dataset to maintain fixed B.

solve $n \cdot \epsilon_{\text{token}} \cdot D_{\max} = B$ to find:

$$\epsilon_{d_\chi} = \epsilon_{\text{token}} = \frac{B}{n \cdot D_{\max}}. \tag{6}$$

Paraphrase. Using the proxy rule $2n/\tau = B$, we set the temperature as:

$$\tau = \frac{2n}{B}. \tag{7}$$

PrivacyRestore & PPFT. PrivacyRestore releases a single fixed-size meta-vector, so the accounting is independent of n. After ℓ_2 normalization, $\|u - u'\|_2 \leq 2$ implies a worst-case log-loss bound of at most 2ϵ.

PPFT releases a sequence of obfuscated slot vectors $\tilde{\mathbf{U}} = [\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_m]$ by adding isotropic ℓ_2-Laplace noise to each slot and applying ℓ_2 renormalization as post-processing. Each slot mechanism admits a metric-DP bound (Appendix G), and if at most s slots differ, the worst-case log-loss scales as $2\epsilon s$ under unit-norm boundedness. For numerical alignment with constant-vector baselines, we calibrate PPFT to the same *per-vector* target B:

$$\epsilon_{\text{PR}} = \epsilon_{\text{PPFT}} = \frac{B}{2} = 75.00. \tag{8}$$

G Theoretical Analysis of PPFT under ℓ_2-Laplace Noise

We analyze PPFT under the exact noise injection procedure described in the main text: slot-wise isotropic ℓ_2-Laplace noise followed by ℓ_2 renormalization as post-processing.

G.1 Mechanism Definition

Let X be an input text and $\mathbf{H} = \text{Enc}(X) \in \mathbb{R}^{n \times d_e}$ contextual token embeddings. Non-overlapping k-pooling yields $m = \lceil n/k \rceil$ slot vectors $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_m]$.

For each slot, we sample isotropic ℓ_2-Laplace noise by drawing a direction uniformly on the unit sphere and a radius from a Gamma distribution (shape d_e, rate ϵ), which is equivalent to the density form $p(\mathbf{n}) \propto \exp(-\epsilon\|\mathbf{n}\|_2)$. We then output the obfuscated embedding via post-processing renormalization:

$$\begin{aligned} \mathbf{y}_j &= \mathbf{u}_j + \mathbf{N}_j, \\ p(\mathbf{y}_j \mid \mathbf{u}_j) &\propto \exp(-\epsilon\|\mathbf{y}_j - \mathbf{u}_j\|_2), \\ \tilde{\mathbf{u}}_j &= \frac{\mathbf{y}_j}{\|\mathbf{y}_j\|_2}. \end{aligned} \tag{9}$$

The full output is $\tilde{\mathbf{U}} = [\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_m]$, and slots are perturbed independently.

G.2 Per-slot Metric-DP Guarantee and Composition

Per-slot metric-DP For any two slot vectors $\mathbf{u}, \mathbf{u}'$ and any measurable set $\mathcal{S}$, the pre-normalization mechanism in Eq. (9) satisfies metric DP:

$$\begin{aligned} &P(\mathbf{y} \in \mathcal{S} \mid \mathbf{u}) \\ &\leq \exp\big(\epsilon\|\mathbf{u} - \mathbf{u}'\|_2\big)\ P(\mathbf{y} \in \mathcal{S} \mid \mathbf{u}'). \end{aligned} \tag{10}$$

Proof. Using $p(\mathbf{y} \mid \mathbf{u}) \propto \exp(-\epsilon\|\mathbf{y} - \mathbf{u}\|_2)$,

$$\ln \frac{p(\mathbf{y} \mid \mathbf{u})}{p(\mathbf{y} \mid \mathbf{u}')} = \epsilon\big(\|\mathbf{y}-\mathbf{u}'\|_2 - \|\mathbf{y}-\mathbf{u}\|_2\big) \leq \epsilon\|\mathbf{u}-\mathbf{u}'\|_2,$$

where the inequality follows from the reverse triangle inequality. □

Post-processing. The renormalization $\tilde{\mathbf{u}} = \mathbf{y}/\|\mathbf{y}\|_2$ is deterministic post-processing, so it does not weaken the above metric-DP guarantee.

Slot-sequence composition bound Because slots are perturbed independently, for two sequences $\mathbf{U}, \mathbf{U}'$ we have:

$$\ln \frac{P(\tilde{\mathbf{U}} \mid \mathbf{U})}{P(\tilde{\mathbf{U}} \mid \mathbf{U}')} \leq \epsilon \sum_{j=1}^{m} \|\mathbf{u}_j - \mathbf{u}'_j\|_2. \tag{11}$$

If at most s slots differ and each slot vector is ℓ_2-bounded so that $\|\mathbf{u}_j - \mathbf{u}'_j\|_2 \leq 2$, then the worst-case log-loss is bounded by $2\epsilon s$.

Implication for budget alignment. In practice, a one-token substitution can affect multiple slots due to contextual encoding, so s may exceed 1. In our budget alignment (Appendix E), we match a per-vector worst-case log-loss target B (i.e., $2\epsilon \leq B$) to ensure numerical comparability with constant-vector baselines, and empirically validate inversion resistance.

H Inverse Attack

Threat model. Following prior work on embedding inversion (Morris et al., 2023; Li et al., 2023), we consider an attacker who observes the representation transmitted by the client (e.g., an embedding, an obfuscated query, or an auxiliary vector) and attempts to reconstruct the user prompt (including privacy-sensitive content) using a generative model. Concretely, we instantiate the attacker as `openai-community/gpt2-medium`, a GPT-2 model (Radford et al., 2019), which is fine-tuned to generate the original text from the observed signal.

Common attacker configuration. Across all methods, we use **GPT2-medium** as the attack model, trained for **20 epochs** with learning rate **1e-5** and batch size **32**. During generation, we use greedy decoding with maximum generation length **256**. The attacker is trained on the corresponding training split and evaluated on the test split.

Attack on PPFT (ours). For PPFT, the attacker operates on the same noisy, pooled embedding representation that is exposed to the server. Specifically, we reuse the encoder and k-pooling module from the Stage 1-aligned LLaMA-1B PPFT model to process the input, producing pooled encoder representations identical to those used by PPFT. These pooled embeddings are then passed through a learnable projection layer that maps them to the input embedding space of GPT2-medium, which serves as the attacker decoder. During attack training, the encoder is kept frozen, while only the projection layer and GPT2-medium are optimized. The attacker is trained end-to-end to perform sequence reconstruction, learning to generate the original prompt text from the observed noisy and pooled embeddings.

Attack on PrivacyRestore.
PrivacyRestore (Zeng et al., 2025) transmits an incomplete user query in which privacy-sensitive spans are removed, together with a *meta vector* that encodes information about the removed spans. To match the inference-time observable interface of PrivacyRestore, our inversion attacker is conditioned on *both* the incomplete query and the corresponding meta vector, and is trained to reconstruct the original full query. Specifically, we encode the masked query with the attacker decoder in the standard autoregressive manner, while a learnable projection layer maps the meta vector to the hidden-state dimension of GPT2-medium and injects it as an auxiliary conditioning signal. We jointly fine-tune GPT2-medium and the projection layer under the common attacker configuration to generate the original prompt text from the observable pair.

Attack on d_χ-privacy and Paraphrase. For d_χ-privacy, the client transmits an obfuscated text query obtained by applying token-level privatization, where each token is replaced by a randomized alternative sampled according to a distance-based distribution in an embedding space (Feyisetan et al., 2020). For Paraphrase, the client transmits a paraphrased version of the original query generated by a client-side model. In both cases, the attacker observes only text and directly uses the garbled or paraphrased query as input context to GPT2-medium, which is then fine-tuned to reconstruct the original prompt text using the same attack training procedure described above.

Evaluation metric. We quantify inversion effectiveness using ROUGE-L as a sequence-level reconstruction metric, measuring similarity between the attack model's generated output and the ground-truth original prompt on the test split. Higher ROUGE-L indicates more successful surface-level reconstruction and thus weaker privacy protection. Attribute-level reconstruction metrics are reported separately to assess the recovery of specific sensitive information.

I Noise-Aware Inverse Attack Training

In this additional experiment, we strengthen the adversary by allowing it to train the inverse attack model on noisy representations. All experiments are conducted on the Pri-DDX dataset. Concretely, we keep the inverse model architecture and training procedure identical to the main inverse-attack setting in Appendix H, but inject the same privacy noise during attacker training (i.e., the attacker is trained with representations perturbed under $\epsilon = 75$). This setting tests whether a noise-aware attacker—one that has access to the defense mechanism and can adapt to it—can substantially improve reconstruction of the original input text.

Quantitative results. Table 8 reports ROUGE-L reconstruction scores as a sequence-level similarity metric across privacy budgets and pooling sizes. Overall, the noise-aware attacker achieves higher ROUGE-L than a noise-unaware attacker,

Pooling size	ϵ=0.01	ϵ=0.46	ϵ=0.86	ϵ=2.01	ϵ=2.29	ϵ=17.2	ϵ=22.93	ϵ=75.0
4	0.02974	0.03045	0.03178	0.03487	0.03373	0.16013	0.24380	**0.43974**
8	0.05506	0.05554	0.05525	0.05920	0.06266	0.09974	0.15784	0.33750
16	0.05039	0.05177	0.04938	0.04910	0.05055	0.14032	0.15935	0.17990

Table 8: Noise-aware inverse attack results (ROUGE-L). The attacker is trained with noisy representations while we report reconstruction quality under different privacy budgets at inference.

Ex.	Ground truth	Reconstruction (blue=same, red=different)
1	A 46-year-old male has a history of chronic pancreatitis, diabetes, obesity, pancreatic cancer in family members. The 46-year-old male presents the symptoms of cough, diarrhea, nausea, pain, pale stools and dark urine, skin lesions, underweight. What is the likely diagnosis?	A 6-year-old woman has a history of smoking, diabetes, high blood pressure, obesity, high cholesterol, high blood pressure, smoking. The 6-year-old woman presents the symptoms of cough, fever, fatigue, pain, shortness of breath, skin lesions. What is the likely diagnosis?
2	A 45-year-old woman has a history of chronic pancreatitis, diabetes, obesity, pancreatic cancer in family members, smoking. The 45-year-old woman presents the symptoms of diarrhea, fatigue, nausea, pain, pale stools and dark urine, skin lesions, underweight. What is the likely diagnosis?	A 22-year-old man has a history of alcohol addiction, smoking, alcohol addiction, heart failure, heart valve issue. The 22-year-old man presents the symptoms of chest pain, shortness of breath, pain, fatigue, shortness of breath with exertion, . . .

Table 9: Qualitative examples for the noise-aware inverse attack. Blue indicates spans that exactly match the original prompt, whereas red indicates mismatched or hallucinated content, including medically salient details.

especially in the weak-noise regime (large ϵ). However, even with noise-aware training, the attacker does not recover the full original text: performance remains low for strong noise (small ϵ), and improvements at the inference-time privacy setting ($\epsilon = 75$) remain far from exact reconstruction. Among pooling strategies, pooling-4 is the most vulnerable (0.4397 at $\epsilon = 75$), pooling-8 is intermediate (0.3375), and pooling-16 is the most robust (0.1799). This trend is consistent with the intuition that larger pooling sizes induce stronger information compression, making exact inversion intrinsically harder even when the attacker matches the training-time noise distribution.

Importantly, we also conducted a matched noise-aware comparison for PrivacyRestore under the same inference-time privacy setting ($\epsilon = 75$). Under this stronger attacker, PrivacyRestore reaches a substantially higher reconstruction score (ROUGE-L up to 0.72), whereas PPFT remains markedly lower across all pooling settings. This comparison is critical because it shows that the stronger attack does not simply increase reconstruction for all methods uniformly; rather, PPFT retains a clear advantage even when the adversary is fully aware of the defense mechanism and trained on noise-corrupted representations.

These results also highlight an important caveat: ROUGE-L can be inflated when the attacker learns to replicate common scaffolding tokens and templates, even if the recovered content is factually inconsistent with the original private text. Therefore, while noise-aware training increases lexical overlap, it does not imply faithful reconstruction. Taken together with the matched PrivacyRestore comparison, our results show that PPFT provides substantially stronger reconstruction resistance under realistic privacy-preserving inference conditions.

Qualitative analysis: template-matching rather than true recovery. Despite higher ROUGE-L at large ϵ, outputs often improve by mimicking the *surface form* of the data (e.g., age/gender template and symptom-list scaffolding), rather than recovering correct patient attributes or medical history. Table 9 provides two representative cases, where tokens identical to the ground truth are highlighted in blue, while mismatched or hallucinated content is highlighted in red. As shown, the attacker frequently reproduces high-frequency structural phrases (e.g., "has a history of", "presents the symptoms of", and the question suffix), yet changes medically salient details such as age, gender, comorbidities, and symptom composition.

J Inversion Attack with a Stage-1 Aligned Model

In this additional setting, we consider a stronger adversary that better reflects a realistic threat model for LLM service providers. Specifically, we assume

Ex.	Ground truth	Reconstruction (blue=same, red=different)
1	A 57-year-old male has a history of antipsychotic medication usage, nausea, stimulant drug use. The 57-year-old male presents the symptoms of involuntary eye movement, jaw pain, muscle spasms, muscle spasms in neck, ptosis, shortness of breath. What is the likely diagnosis?	The diagnosis of the 57-year-old malewho has been experiencing symptoms of eye jumping, unknown button, joint pain and muscle spasms in neck, is psychosis. What is the diagnosis?
2	A 8-year-old woman has a history of active cancer, deep vein thrombosis, hormone intake, immobility for >3 days, surgery within last month. The 8-year-old woman presents the symptoms of coughing up blood, loss of consciousness, pain, shortness of breath, swelling. What is the likely diagnosis?	The patient has been in the hospital for over 3 weeks, with intravenous drug use, migraine, intake of bed, surgery. The patient's symptoms are cough, fever, pain, swelling. What is the likely diagnosis?

Table 10: Qualitative examples for the Stage-1 aligned inversion attacker. Blue spans exactly match the original prompt, while red spans differ. Even with a stronger attacker aligned to the encoder space, reconstructions often preserve only partial lexical overlaps rather than medically faithful recovery.

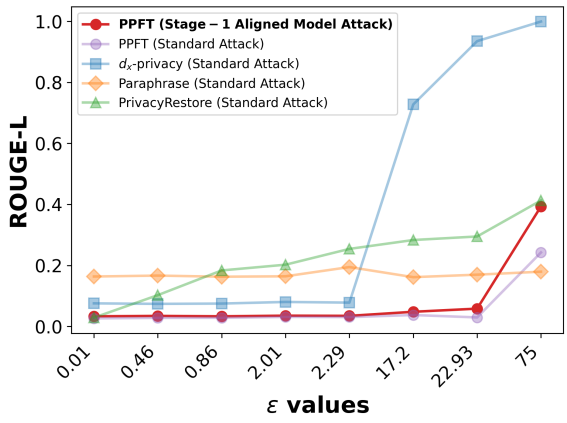

Figure 6: Inversion attacks on PPFT using a Stage-1 aligned model (stronger attacker) under varying privacy budgets ϵ. For comparison, we also report inversion results from a GPT-2 Medium model (weaker attacker).

the provider is willing to recover user prompts and thus replaces the inversion attacker (GPT-2 Medium in Appendix H) with a *Stage-1 aligned model*—i.e., a decoder already aligned to the encoder representations during Stage 1. This attacker starts from a substantially more favorable initialization since it has been explicitly trained to interpret the encoder-aligned latent space. All other training and evaluation conditions follow Appendix H.

Quantitative results. Figure 6 reports ROUGE-L reconstruction scores across privacy budgets. While the Stage-1 aligned attacker slightly improves reconstruction quality in the weak-noise regime, it still fails to faithfully recover the original prompt. Notably, under the inference-time condition ($\epsilon = 75.0$), ROUGE-L reaches 0.393, remaining below 0.4.

Qualitative analysis. Table 10 shows representative reconstructions. Spans that *exactly match* the original prompt are highlighted in blue, whereas altered or hallucinated content is highlighted in red. Even with the Stage-1 aligned attacker, improvements in ROUGE-L largely come from reproducing a subset of frequent tokens or local phrases, while medically salient attributes (e.g., history and symptom composition) are not reliably recovered.

K Universal Zero-shot Embedding Inversion under Token Pooling

Recent work has shown that text embeddings can be inverted to recover substantial semantic information about the original inputs, even under black-box access assumptions (Morris et al., 2023; Zhang et al., 2025). These attacks, however, are primarily studied under encoders that map an entire input sequence to a *single* embedding vector. In this appendix, we examine whether such inversion techniques remain effective when the encoder employs *token pooling*, producing multiple embeddings per input.

Threat Model. We consider a black-box adversary who has access to (i) the pooled embeddings of a private input and (ii) query access to the same encoder used to generate those embeddings. This setting is consistent with prior embedding inversion work (Morris et al., 2023; Zhang et al., 2025), but differs in that the encoder applies pooling over fixed-size token blocks (k=4 in our experiments), followed by noise injection. The adversary attempts to reconstruct the original text using iterative, embedding-guided decoding.

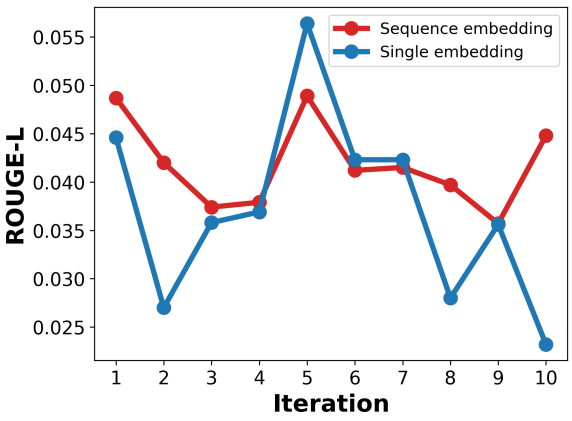

Figure 7: Reconstruction quality across iterations for the pooled-embedding (Experiment 1) and single-embedding (Experiment 2) settings.

Experimental Setup. We conduct two inversion experiments on the Pri-NLICE dataset introduced by Zeng et al. (2025). In both cases, the target encoder is a LoRA-adapted `Llama-3.2-1B-Instruct` model with pooling size k=4 and Laplace noise injection (ϵ=75). For generation, we use `meta-llama/Llama-3.2-3B-Instruct` as the decoder. To ensure a fair comparison, we use the same privacy parameter ϵ for inversion experiments as in the inference setting reported in Table 1.

Following the adversarial decoding paradigm of Zhang et al. (2025), we perform iterative inversion for up to 10 iterations. At each iteration, the decoder generates candidate texts using embedding-guided search, and the highest-scoring candidate (based on cosine similarity in embedding space) is selected and used as the seed for the next iteration. Reconstruction quality is evaluated using ROUGE-L against the ground-truth text, averaged over the dataset.

Experiment 1: Pooling-Aligned Inversion. In the first experiment, we directly attack the pooled representation. The encoder outputs a *sequence of pooled embeddings* (one per 4 tokens), and during inversion we compute cosine similarity block-wise between generated and target embeddings, aggregating scores across aligned blocks. Generation is constrained to the original input length, ensuring that the number of pooled embeddings in the generated text does not exceed that of the target. Figure 7 reports ROUGE-L scores across iterations.

Despite iterative refinement, reconstruction quality remains low and does not exhibit a consistent upward trend. This contrasts sharply with prior results on non-pooled encoders, where repeated iterations significantly improve lexical overlap (Zhang et al., 2025).

Experiment 2: Mean-Pooled Single-Vector Inversion. To more closely match the setting of prior work, we perform a second experiment in which the pooled embeddings are averaged into a single vector after noise injection. This removes the structural mismatch between pooled encoders and single-vector inversion methods. Since the target representation is now a single embedding, we allow the decoder to generate up to 250 tokens, mirroring the unconstrained generation setting used in Zhang et al. (2025). Figure 7 reports ROUGE-L scores across iterations.

Although this setting removes the pooling mismatch, inversion performance remains poor. Even at its peak (iteration 5), ROUGE-L remains below 0.06, and later iterations often degrade reconstruction quality.

Discussion. Across both experiments, embedding inversion fails to recover meaningful lexical information from pooled, noise-injected embeddings. This is notable because the second experiment explicitly aligns with the assumptions of prior inversion attacks by collapsing the pooled representation into a single embedding. The results suggest that the combination of token pooling and noise injection substantially alters the embedding landscape, making iterative, cosine-similarity-guided decoding ineffective.

From a security perspective, these findings indicate that pooling-based encoders provide a qualitatively stronger defense against embedding inversion than previously studied single-vector encoders. In contrast to earlier conclusions that "embeddings reveal (almost) as much as text" (Morris et al., 2023), our results show that this claim does not directly extend to encoders that disrupt token-level alignment through pooling.

L Server-Side Restoration-Instruction Attack

In this additional experiment, we study whether a malicious server can coerce the decoder to reconstruct the client's original prompt by injecting an explicit restoration instruction. Unlike standard prompt injection in text-based interfaces, the attacker never observes the raw user text and only

Pooling size	Dataset	ROUGE-L	BLEU
pool4	Pri-DDX	0.0496	0.0058
pool4	Pri-NLICE	0.0378	0.0112
pool4	Pri-SLJA	0.0335	0.0029
pool8	Pri-DDX	0.0459	0.0066
pool8	Pri-NLICE	0.0592	0.0217
pool8	Pri-SLJA	0.0437	0.0028

Table 11: Results of the server-side restoration-instruction attack with Llama-3.1-8B-Instruct. Lower ROUGE-L and BLEU indicate stronger resistance to server-side restoration attempts.

receives the client-transmitted embedding representation. The attacker then adds a server-side instruction such as *Restore the original text.* to test whether the decoder can be repurposed as a text restorer under the PPFT interface.

Experimental Setup. We evaluate a setting in which the user-side query signal and the server-side restoration instruction are provided through separate input paths. The user input is never given as plain text; instead, the client sends only pooled embedding representations. The attacker injects the restoration instruction on the server side while attempting to reconstruct the original text from the observed embedding input. Since the attacker does not know the client's noise process, we provide only the encoder outputs with the same pooling operation applied, without noise injection. We report mean ROUGE-L and mean BLEU on Pri-DDX, Pri-NLICE, and Pri-SLJA using Llama-3.1-8B-Instruct. The results are summarized in Table 11.

Results and Discussion. Across all datasets and models, reconstruction quality remains very low, with mean ROUGE-L staying around 0.03–0.06 and mean BLEU remaining close to zero. These results indicate that, even when the server injects an explicit restoration-oriented instruction, the decoder cannot faithfully recover the original text from the transmitted input representation. This suggests that the difficulty of restoration does not primarily come from the instruction format itself, but from the PPFT design in which the server receives embeddings rather than the raw text. Overall, the results provide additional evidence that transmitting pooled embeddings instead of the original prompt substantially limits server-side attempts to reconstruct private user inputs.