Ask in the Crowd: Differentially Private LLM Inference via Dummy-Augmented Shuffling

Zhihao Liu[1*] **Zixiong Guo**[1*] **Shuo Shao**[1] **Yu He**[1] **Wenli Wang**[1]

Meihui Chen[2†] **Di Wang**[3] **Yuke Hu**[3†]

[1]Zhejiang University [2]Xiaomi Corporation [3]King Abdullah University of Science and Technology

{zhihao_liu, guozixiong, shaoshuo_ss, yuherin, wangwenli}@zju.edu.cn

chenmeihui@xiaomi.com {di.wang, yuke.hu}@kaust.edu.sa

Abstract

Large language models are increasingly deployed as LLM-as-a-Service (LLMaaS), where user queries are sent to remote LLMs for inference, raising severe privacy concerns about the exposure of sensitive user inputs. Differentially private inference mitigates these risks by injecting noise into input embeddings. However, existing methods struggle to balance privacy and utility, as stronger privacy protection often leads to substantial accuracy degradation in LLM inference.

In this work, we propose DP-SPD, a differentially private LLM inference framework that protects user queries by hiding the real query among client-generated dummy queries. All queries are randomly shuffled before being sent to the server, preventing direct identification of the user's true intent. To improve inference efficiency, we exploit the shared-prefix structure among queries and reuse transformer key–value (KV) caches during decoding. We analyze the privacy guarantee of DP-SPD and evaluate its utility on classification and generation tasks, as well as its privacy protection under QIA, EIA, and GPT-based attacks. Experimental results show that our method achieves strong inference-time privacy protection and near-plaintext utility while improving inference efficiency through shared-prefix KV-cache reuse.

1 Introduction

Large language models (LLMs) are increasingly offered as LLM-as-a-Service (LLMaaS) and deployed on cloud-based platforms [1, 2, 3, 4], where user queries are transmitted to remote servers for inference, enabling scalable and high-quality language understanding. However, this deployment paradigm also raises serious privacy concerns [5, 6]: user inputs may contain private information, such as medical records, health conditions, financial details, location traces, or private communications, which are directly exposed to the LLM service provider. Such exposure risks data leakage, misuse, or memorization by the model, making privacy-preserving inference essential for safe and trustworthy LLMaaS deployment.

However, existing approaches to privacy-preserving LLM inference face a difficult trade-off among privacy, utility, and efficiency. Cryptography-based methods such as secure computation [7, 8] and homomorphic encryption [9, 10] which provide strong privacy guarantees by design, as the LLM server never observes raw user inputs. However, their substantial computational and communication overhead often limits their practicality in large-scale LLMaaS deployments. Perturbation-based

*These authors contributed equally to this work.

†Corresponding authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

DP methods [11, 12, 13, 14, 15, 16] incur lower computational overhead, but injecting random perturbations into tokens or embeddings inevitably degrades output quality [13, 15], while denoising-based methods introduce additional inference-time cost [12]. As a result, existing DP LLM inference approaches still face a practical challenge in balancing privacy, utility, and efficiency.

This limitation motivates us to move beyond relying solely on stronger perturbation and instead introduce anonymity into private LLM inference. Specifically, we explore client-side dummy augmentation, where the real query is hidden among multiple plausible dummy queries before being sent to the server. In this way, the server no longer observes a single identifiable request, but a shuffled set of hard-to-distinguish candidate queries. However, unlike classical shuffle DP, LLM inference usually involves a single user query. Therefore, the client must locally construct the anonymity set rather than relying on reports from other independent users. Such client-side construction introduces two challenges: ① dummy queries must be contextually plausible enough to avoid being filtered by the server, and ② dummy-augmented inference must remain efficient enough for practical use.

In this paper, we aim to improve the "privacy–utility–efficiency" trade-off for differentially private LLM inference. We propose a shuffle-based privacy-preserving inference framework DP-SPD that hides the real user query among dummy queries. DP-SPD first selects and privatizes a shared prefix in the continuous embedding space under metric DP. A lightweight client-side continuation model then generates prefix-consistent dummy suffixes from the privatized prefix, providing stronger camouflage for the real continuation. Finally, the real and dummy inputs are randomly shuffled on the client side before being sent to the server. Furthermore, by exploiting the shared-prefix structure together with transformer key–value (KV) cache reuse, DP-SPD substantially reduces the overhead of dummy-augmented inference. Our main contributions are summarized as follows:

- We propose a shuffle-based privacy-preserving inference framework DP-SPD that combines prefix-consistent dummy suffix generation with client-side shuffling. By sharing prefixes across real and dummy queries, DP-SPD enables KV-cache reuse and reduces redundant inference overhead.
- We provide rigorous privacy and correctness analyses for the framework, formally establishing its guarantees under d_χ-privacy and proving that the client can reliably identify the output corresponding to the real input.
- We conduct extensive experiments on multiple models and tasks to evaluate utility and inference efficiency. We also evaluate privacy leakage under query identification attack, embedding inversion attack and GPT-based attack to demonstrate that DP-SPD achieves a stronger privacy–utility–efficiency trade-off than existing perturbation-based baselines.

2 Background and Related Work

2.1 Inference-Time Privacy Risks in LLMaaS

Large Language Models (LLMs) are increasingly deployed as LLM-as-a-Service (LLMaaS) [1, 2, 3, 4, 17], where users access cloud-hosted models through black-box APIs without visibility into model parameters or inference internals. This deployment paradigm enables scalable and cost-efficient access to foundation models, but fundamentally shifts inference to a centralized setting. However, LLMaaS requires transmitting user prompts verbatim to centralized servers, exposing sensitive inference-time inputs to the service provider. Prior work [18, 19] has demonstrated that such inputs may be persistently logged, implicitly memorized by models, or indirectly leaked through system-level channels and auxiliary artifacts, raising serious concerns about the privacy of user interactions. As a result, such privacy risks are particularly concerning for real-world LLMaaS deployments involving sensitive or proprietary content.

2.2 Differentially Private LLM Inference

To address inference-time privacy risks in LLMaaS, perturbation-based mechanisms have emerged as a practical approach for privacy-preserving LLM inference. DP-Forward [11] perturbs input embeddings and forwards the noisy representations directly to the LLM for inference. Split-and-Denoise (SnD) [12] further uses an auxiliary denoising module to recover semantic information from the perturbed embeddings. In contrast, InferDPT [13] introduces a prompt-level local differential privacy mechanism combined with response extraction strategies. ObfusLM [14] operates in the

embedding space and employs obfuscation and shuffling to defend against inversion and query identification attacks. Despite these advances, the inherent characteristics of LLM inference, most notably the **single-query** execution paradigm, fundamentally constrain the effective use of differential privacy amplification. Beyond this core challenge, existing methods suffer from additional drawbacks, including reliance on extra denoising models and increased semantic distortion caused by discrete token perturbation. As a result, achieving a balanced "privacy–utility–efficiency" trade-off for differentially private LLM inference remains an open problem.

3 Preliminary

3.1 Differential Privacy

Differential privacy (DP) [20] is a standard formal notion of data privacy, defined as follows:

Definition 3.1 ((ϵ, δ)-Differential Privacy)**.** A randomized mechanism M with domain D and range R preserves (ϵ, δ)-differential privacy if for any two neighboring datasets $D, D' \in D$ and for any subset $S \subseteq R$, the following inequality holds:

$$\Pr[M(D) \in S] \leq e^{\epsilon} \Pr[M(D') \in S] + \delta \tag{1}$$

where ϵ is the privacy budget and δ is the failure probability.

Local differential privacy (LDP) is a client-side privacy model in which each user locally perturbs their input before sending it to an untrusted server.

Definition 3.2 (ϵ-Local Differential Privacy)**.** A randomized mechanism $M : \mathcal{X} \to \mathcal{Y}$ satisfies ϵ-local differential privacy if for any two inputs $x, x' \in \mathcal{X}$ and for any measurable set $S \subseteq \mathcal{Y}$,

$$\Pr[M(x) \in S] \leq e^{\epsilon} \Pr[M(x') \in S]. \tag{2}$$

Definition 3.3 (d_χ-Privacy)**.** Let $(\mathcal{X}, d_\chi)$ be a metric space where d_χ denotes a distance metric (e.g., Euclidean distance). A randomized mechanism $M : \mathcal{X} \to \mathcal{Y}$ satisfies ϵ-d_χ-privacy if for any $x, x' \in \mathcal{X}$ and any $S \subseteq \mathcal{Y}$:

$$\Pr[M(x) \in S] \leq e^{\epsilon \cdot d_\chi(x, x')} \Pr[M(x') \in S]. \tag{3}$$

Given a tokenized query sequence $X = [x_1, \ldots, x_n]$, the embedding layer maps it to $\mathbf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{n \times d}$ via an embedding matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $|\mathcal{V}|$ is the vocabulary size and d is the embedding dimension. Under the d_χ-privacy metric with parameter ϵ and L_2 distance, privacy is enforced by adding Laplacian noise $\boldsymbol{z}$ with density proportional to $\exp(-\epsilon\|z\|)$ [21]. Specifically, the noise is sampled as $\boldsymbol{z} = lv$, where $l \sim \Gamma(d, 1/\epsilon)$ and v is uniformly drawn from the unit ball B^d, yielding the privatized embedding $M(\boldsymbol{x}_t) = \boldsymbol{x}_t + \boldsymbol{z}$.

Furthermore, to mitigate the utility degradation caused by local noise, we operate under the shuffle model of differential privacy [22, 23, 24].

Definition 3.4 (Shuffle Model)**.** Consider k users, each holding a private input x_i. The Shuffle Model consists of a local randomizer $\mathcal{R}$ and a trusted shuffler $\mathcal{A}$. Each user applies $y_i = \mathcal{R}(x_i)$, and the shuffler applies a uniform random permutation π to the batch of messages. The server observes the multiset:

$$\mathcal{M}_{shuffle}(x_1, \ldots, x_k) = \{y_{\pi(1)}, y_{\pi(2)}, \ldots, y_{\pi(k)}\}. \tag{4}$$

The core advantage of this model is *Privacy Amplification*: shuffling provides a stronger central DP guarantee than the local guarantee of $\mathcal{R}$ alone, since the adversary cannot easily link a specific output message y_j back to a specific user i.

3.2 KV-cache in LLM Inference

To mitigate the computational redundancy of autoregressive generation, LLMs employ the *KV-cache* mechanism [25, 26]. Rather than recomputing the Key (K) and Value (V) representations for the entire context at each decoding step, the model caches these intermediate states from previous tokens. At step i, the model computes only the current query, key, and value vectors (q_i, k_i, v_i), attends q_i over the cached keys and values, and then appends the new (k_i, v_i) pair to the cache. This design substantially improves inference efficiency, at the cost of maintaining a persistent cache whose size grows linearly with the sequence length.

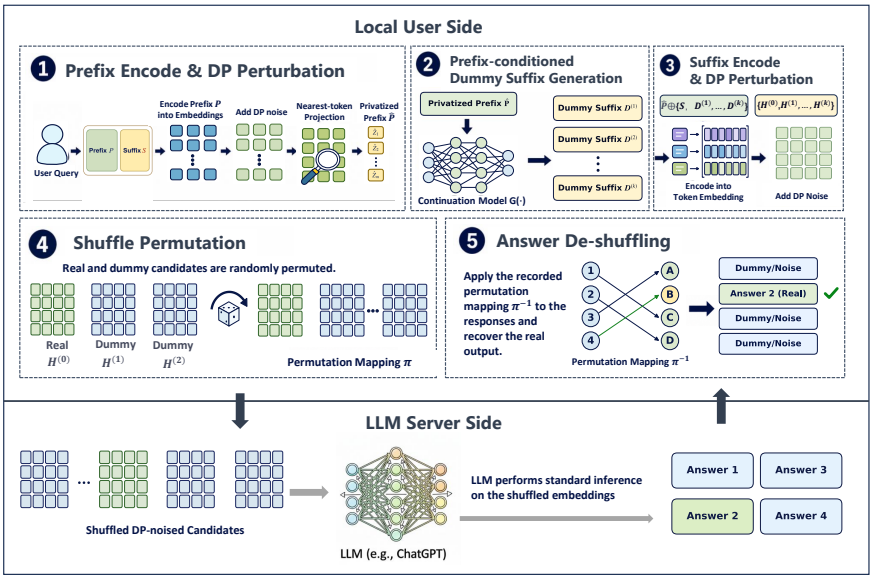


Figure 1: Full workflow of DP-SPD. The client generates prefix-consistent dummy suffixes, perturbs and shuffles them locally, and sends the shuffled noisy candidates to the server for inference.

4 Methodology

We present **D**ifferentially **P**rivate **S**huffling with **P**refix-conditioned **D**ummies (DP-SPD), a privacy-preserving inference framework for LLMs. Given a user query, the client generates multiple dummy suffixes conditioned on the same prefix, locally perturbs all candidate suffixes, and randomly shuffles them before sending them to the server. The shared-prefix design further enables KV-cache reuse, reducing the inference overhead introduced by dummy augmentation.

4.1 DP-SPD Mechanism

Our design is motivated by two properties of LLM inference that make standard privacy mechanisms difficult to apply directly.

① Token-level differences do not faithfully reflect **semantic distance**. Queries with many token changes may remain semantically close, while small token changes can substantially alter the meaning. Thus, standard local differential privacy can require overly strong perturbation and hurt utility. We instead adopt **Metric Differential Privacy** (d_χ-privacy) [27, 28], which measures privacy loss by embedding distance and better captures semantic privacy in LLM inference.

② Shuffling cannot be directly applied to **single-user LLM inference**. Classical privacy amplification by shuffling relies on reports collected from multiple independent users, where anonymity comes from hiding which randomized report belongs to which user. In LLM inference, however, the interaction is typically initiated by a single user. A natural remedy is to introduce synthetic dummy inputs on the client side, but naive dummy construction leads to a tension between privacy and efficiency. Sentence-level random sampling can create full dummy queries that are less trivially tied to the real input, but each dummy query requires a separate forward pass, substantially increasing the **inference overhead**. Token-level random sampling allows prefix sharing and **KV-cache** reuse, but such token-level dummies are often contextually or **semantically inconsistent** with the real query. As a result, the server may distinguish them using contextual consistency, semantic relevance, or prior knowledge, causing the effective anonymity set to collapse.

DP-SPD addresses these issues by constructing dummies under the same shared prefix as the real query. Prefix conditioning makes the dummy suffixes more context-compatible with the real suffix, so that the shuffled candidates form a more meaningful anonymity set. At the same time, local perturbation provides a metric-DP guarantee over each candidate, while shuffling the real and dummy suffixes further obscures which perturbed suffix corresponds to the user's true private input. Thus, DP-

Algorithm 1 DP-SPD Main Pipeline

Require: Tokenized query $X = [x_1, \ldots, x_n]$, continuation model G, embedding function E, projection operator $\Pi_{\mathcal{V}}$, dummy ratio r, dummy number k, privacy budgets $\epsilon_{\text{pre}}, \epsilon_{\text{suf}}$, deployment mode m

Ensure: Server responses $\mathcal{Y}$ and real-query output y_{real}

$T \leftarrow \lfloor (1-r)n \rfloor$
$\mathcal{P} \leftarrow [x_1, \ldots, x_T], \quad \mathcal{S} \leftarrow [x_{T+1}, \ldots, x_n]$
$(\widetilde{\mathcal{P}}_{\text{emb}}, \widehat{\mathcal{P}}) \leftarrow \textsc{PrivatizePrefix}(\mathcal{P}, E, \Pi_{\mathcal{V}}, \epsilon_{\text{pre}})$
$\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)} \sim G(\widehat{\mathcal{P}}, n-T)$
$H^{(0)}, \ldots, H^{(k)} \leftarrow \textsc{BuildCandidates}(\widetilde{\mathcal{P}}_{\text{emb}}, \mathcal{S}, \{\mathcal{D}^{(i)}\}_{i=1}^{k}, E, \epsilon_{\text{suf}})$
Sample a uniform random permutation π on $\{0, \ldots, k\}$
$H \leftarrow [H^{(\pi(0))}, H^{(\pi(1))}, \ldots, H^{(\pi(k))}], \quad j \leftarrow \pi^{-1}(0)$
if $m =$ `api-only` **then**
  $\widehat{H} \leftarrow \textsc{ProjectBatch}(H, E, \Pi_{\mathcal{V}})$
  $\mathcal{Y} \leftarrow \mathcal{F}_{\text{api}}(\widehat{H})$
else
  $\mathcal{Y} \leftarrow \mathcal{F}_{\text{embed}}(H)$
end if
$y_{\text{real}} \leftarrow \mathcal{Y}[j]$

SPD combines metric-based local perturbation, prefix-consistent dummy construction, and client-side shuffling into a unified framework for practical privacy-preserving LLM inference.

As shown in Figure 1 and Algorithm 1, given a query $X = [x_1, \ldots, x_n]$ and a dummy ratio $r \in [0, 1]$, let $T = \lfloor (1-r)n \rfloor$, DP-SPD first splits it into a prefix $\mathcal{P} = [x_1, \ldots, x_T]$ and a suffix $\mathcal{S} = [x_{T+1}, \ldots, x_n]$. The client then perturbs the prefix embeddings with privacy budget ϵ_{pre} to preserve prefix alignment and prevent averaging attacks, and then projects the noisy prefix embeddings back to discrete tokens, obtaining a prefix $\widehat{\mathcal{P}}$. The continuation model G uses $\widehat{\mathcal{P}}$ to generate k dummy suffixes:

$$\mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)} \sim G(\widehat{\mathcal{P}}, n-T).$$

The candidate suffix set is then

$$\mathbb{C} = \{\mathcal{S}, \mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)}\}.$$

For $t > T$, the real suffix and each dummy suffix are perturbed independently in BUILDCANDIDATES, where $\mathbf{z}_{0,t}, \mathbf{z}_{i,t} \sim \text{Lap}(1/\epsilon_{\text{suf}})$. The client then obtains $k+1$ noisy candidates $\{H^{(0)}, \ldots, H^{(k)}\}$, samples a random permutation π, and sends the shuffled batch to the server, while recording $j = \pi^{-1}(0)$. Since all candidates share the same prefix, the server can reuse KV caches for that prefix and only compute distinct suffixes. Depending on the deployment mode, the shuffled batch is either sent directly as embeddings to a fully accessible LLM server, or first discretized by nearest-token projection and then sent as token sequences to a black-box API. The auxiliary routines PRIVATIZEPREFIX, BUILDCANDIDATES, and PROJECTTOTOKENS are provided in Appendix A.

4.2 Privacy Analysis of DP-SPD

The overall privacy guarantee of DP-SPD consists of two parts: the privacy contribution of the shared prefix and the suffix candidate set, which contains the real suffix together with the dummy suffixes. The shared prefix is locally perturbed, but it does not benefit from suffix-level shuffling amplification. In contrast, the real suffix is perturbed and then hidden among k dummy suffixes through client-side shuffling. We therefore analyze these two components separately.

Theorem 4.1 (Privacy Analysis for Prefix)**.** *Let $\mathcal{R}_{\text{pre}}$ be the local randomizer applied to the shared prefix, and suppose that $\mathcal{R}_{\text{pre}}$ satisfies ϵ_{pre}-d_χ-privacy. The privatized prefix $\widehat{\mathcal{P}} = \mathcal{R}_{\text{pre}}(\mathcal{P})$ satisfies ϵ_{pre}-d_χ-privacy. Define the exposure of $\mathcal{P}$ as*

$$\text{Exp}(\mathcal{P}) = \{\widehat{\mathcal{P}}, \mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)}\},$$

where $D^{(i)} \sim G(\widehat{P}, n-T)$. Then $\text{Exp}(\mathcal{P})$ also satisfies ϵ_{pre}-d_χ-privacy.

Proof. Since G is a randomized post-processing of $\widehat{\mathcal{P}}$, the output of $\mathcal{R}_{\mathrm{pre}}$, the dummy generation step does not directly access the raw prefix $\mathcal{P}$. The exposure

$$\mathrm{Exp}(\mathcal{P}) = \{\widehat{\mathcal{P}}, \mathcal{D}^{(1)}, \ldots, \mathcal{D}^{(k)}\}, \qquad \mathcal{D}^{(i)} \sim G(\widehat{\mathcal{P}}, n-T),$$

is therefore a randomized post-processing of $\widehat{\mathcal{P}} = \mathcal{R}_{\mathrm{pre}}(\mathcal{P})$. By the post-processing property of differential privacy [29], this transformation cannot increase the privacy loss. Hence, $\mathrm{Exp}(\mathcal{P})$ satisfies the same privacy guarantee as $\mathcal{R}_{\mathrm{pre}}$. □

We next analyze the suffix part. Fix a privatized prefix $\widehat{\mathcal{P}}$. Let $\mathcal{S}$ and $\mathcal{S}'$ be two possible real suffixes under the same prefix, and let $\boldsymbol{s}$ and $\boldsymbol{s}'$ denote their embedding representations. We define their suffix distance as

$$c = \|\boldsymbol{s} - \boldsymbol{s}'\|.$$

Let $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_k$ denote the dummy suffix embeddings generated from the same privatized prefix $\widehat{\mathcal{P}}$. The two suffix candidate batches are

$$\mathcal{S} = \{\boldsymbol{s}, \boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\}, \qquad \mathcal{S}' = \{\boldsymbol{s}', \boldsymbol{d}_1, \ldots, \boldsymbol{d}_k\}.$$

Thus, the two batches differ only in the real suffix embedding, while the dummy suffix embeddings follow the same distribution.

Theorem 4.2 (Privacy Analysis for Suffix)**.** *Suppose the local randomizer on suffix embeddings satisfies ϵ-d_χ-privacy. Then, for any measurable output set $\mathcal{A}$ and any suffix pair $(\mathcal{S}, \mathcal{S}')$ with distance $c = \|\boldsymbol{s} - \boldsymbol{s}'\|$, DP-SPD satisfies ϵ''-d_χ-privacy with probability at least $1-\delta$, where*

$$\epsilon'' = \frac{1}{c} \log\left(1 + \frac{e^{c\epsilon} - 1}{e^{c\epsilon} + 1}\left(\frac{8\sqrt{e^\epsilon \log(4/\delta)}}{\sqrt{k+1}} + \frac{8e^\epsilon}{k+1}\right)\right). \tag{5}$$

The detailed proof of Theorem 4.2 is provided in Appendix C.

Remark 4.3. It is worth emphasizing that despite the difference in privacy definitions, the shuffling amplification under d_χ-privacy preserves the same order as in LDP-based analyses. In particular, the effective privacy loss scales as $\mathcal{O}\big(1/\sqrt{k+1}\big)$ with respect to the number of dummies.

We next establish the privacy guarantee for the complete query. Under our neighboring-query relation, two queries differ either in the prefix or in the suffix.

Theorem 4.4 (Privacy Guarantee for the Complete Query)**.** *Let $\mathcal{Q} = (\mathcal{P}, \mathcal{S})$ and $\mathcal{Q}' = (\mathcal{P}', \mathcal{S}')$ be neighboring queries that differ either in the prefix or in the suffix. Suppose that the prefix mechanism satisfies ϵ_{pre}-d_χ-privacy and that, for every fixed privatized prefix, the suffix mechanism satisfies ϵ_{suf}-d_χ-privacy with probability at least $1-\delta$ in Theorem 4.2. Then, for any output set $\mathcal{A}$,*

$$\Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}) \in \mathcal{A}] \le e^{\epsilon_{\mathrm{qry}} d_\chi(\mathcal{Q}, \mathcal{Q}')} \Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}') \in \mathcal{A}] + \delta,$$

where $\epsilon_{\mathrm{qry}} = \max\{\epsilon_{\mathrm{pre}}, \epsilon_{\mathrm{suf}}\}$.

4.3 Correctness of Answer De-Shuffling

When the server performs inference on a shuffled batch containing both real and dummy inputs, DP-SPD guarantees correct answer de-shuffling, enabling reliable recovery of the correct answer.

Theorem 4.5 (Correctness of Answer De-Shuffling)**.** *Let f be the server-side inference function and let π be the client-side permutation. Given the shuffled batch*

$$H = \big[H^{(\pi(0))}, H^{(\pi(1))}, \ldots, H^{(\pi(k))}\big],$$

the client can uniquely recover the output of the real query.

Proof. Since the permutation π is generated by the user, the user stores the index j such that $j = \pi^{-1}(0)$. After the server returns

$$\mathcal{Y} = \{f(H^{(\pi(0))}), \ldots, f(H^{(\pi(k))})\},$$

the client retrieves $\mathcal{Y}[j] = f(H^{(0)})$, ensuring correct answer de-shuffling. □

5 Experiments

5.1 Experiment Setups

Models and Datasets. For classification, LLaMA-2-7B-Chat [30] is evaluated on the IMDb movie review dataset [31], while `bert-base-uncased` [32] is fine-tuned and evaluated on SST-2 and QNLI from the GLUE benchmark [33]. We report classification accuracy. For generation, we evaluate LLaMA-2-7B-Chat on WikiText-2 [34], Stanford Alpaca [35], and Databricks-Dolly-15k [36]. To examine whether DP-SPD generalizes to stronger models and more challenging tasks, we additionally evaluate Qwen3-14B [37] on CNN/DailyMail summarization [38, 39], reporting perplexity (PPL) [40] and ROUGE-L [41]. We further evaluate Qwen3-32B on 1,000 real-world queries from WildChat [42], comprising 500 answerable QA requests and 500 open-ended requests. We report QA accuracy and open-ended response quality on a 1–5 scale. Dummy suffixes are generated by a lightweight client-side continuation model. We use Qwen2.5-0.5B-Instruct [43] for this purpose.

Metrics and baselines. To directly measure privacy leakage, we additionally evaluate DP-SPD under Query Identification Attack, Embedding Inversion Attack [44] and an LLM-assisted attack inspired by the GPT inference attack [13], where GPT-4o-mini [45] is used to identify the real candidate and recover the real suffix. We report attack success rate (ASR), real-candidate PPL, dummy-candidate PPL, and the PPL gap between real and dummy candidates for QIA and others.We compare against DP-Forward [11], CAPE [46], TextObfuscator [47], SANTEXT+ [48], CUSTEXT+ [49], SnD [12], and InferDPT [13]; incompatible pipelines are implemented as *style adapters* preserving their core privacy mechanisms.

5.2 Utility Evaluation

5.2.1 Classification Task

To evaluate whether DP-SPD preserves downstream utility across different model architectures and task settings, we conduct classification experiments with BERT-base on SST-2 and QNLI and with LLaMA-2-7B-Chat on IMDb. Besides comparing against representative privacy-preserving baselines, we study how the anonymity-set size k and dummy ratio r affect utility under different privacy budgets. We make the following key observations.

DP-SPD Preserves Near-Plaintext Classification Utility. Table 1 shows that conventional perturbation-based baselines suffer substantial utility degradation, especially on QNLI: DP-Forward is close to random guessing, while CAPE, TextObfuscator, and other token- or representation-perturbation methods remain well below plaintext inference. In contrast, both DP-SPD variants retain above 90% accuracy on SST-2 and QNLI and remain within roughly one percentage point of the plaintext baseline. The API-only variant performs comparably to the embedding-access setting, indicating that nearest-token projection introduces little additional utility loss. These results suggest that DP-SPD preserves task-relevant information more effectively than directly perturbing tokens or intermediate representations, without requiring an additional denoising model at inference.

Table 1: Classification accuracy (%) on SST-2 and QNLI using BERT-base at $\epsilon = 3$. "Plaintext" denotes raw-input inference.

Method	SST-2	QNLI
Plaintext	92.02	91.40
DP-Forward [11]	51.69	49.48
CAPE [46]	76.54	52.02
TextObf. [47]	79.93	58.68
SANTEXT+ [48]	83.71	81.64
CUSTEXT+ [49]	79.12	77.75
SnD [12]	72.90	63.20
DP-SPD (Embedding)	**91.06**	**90.36**
DP-SPD (API-only)	**90.60**	**91.31**

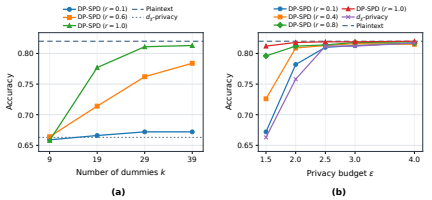


Figure 2: Classification utility on IMDb using LLaMA-2-7B-Chat. (a) Accuracy versus k at $\epsilon = 1.5$. (b) Accuracy across privacy budgets with $k = 29$.

Larger Anonymity Sets Help Most under Tight Privacy. Figure 2(a) shows that increasing k is particularly beneficial when the privacy budget is small. At $\epsilon = 1.5$ and $r = 0.6$, increasing k from 9 to 39 raises IMDb accuracy from 66.4% to 78.4%. The gain also depends strongly on the dummy ratio: at the same privacy budget, increasing k provides only a small improvement when $r = 0.1$, but a much larger gain for $r = 0.6$ and $r = 1.0$. This behavior is consistent with shuffling amplification:

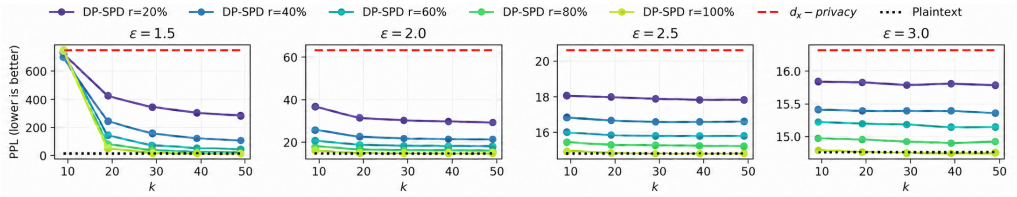


Figure 4: Generation utility on WikiText-2 with LLaMA-2-7B-Chat. Lower PPL is better. We vary the privacy budget ϵ, dummy number k, and dummy ratio r; larger anonymity sets provide the largest utility gains under tighter privacy.

when a larger portion of the query participates in the shuffled candidate set, enlarging that set allows the mechanism to operate with less perturbation.

Protecting a Larger Suffix Further Improves Utility. Figure 2(b) further shows that increasing the dummy ratio r generally improves utility across privacy budgets. At $\epsilon = 1.5$ and $k = 29$, accuracy rises from 67.2% at $r = 0.1$ to 81.2% at $r = 1.0$, approaching the 82.0% plaintext result. The difference becomes progressively smaller as ϵ increases, indicating that the benefit of a larger suffix is most important in the high-privacy regime. Taken together, the two ablations show that k and r provide complementary controls over the privacy–utility trade-off: larger k strengthens the anonymity set, while larger r allows a larger fraction of the query to benefit from shuffling amplification.

5.2.2 Generation Task

To evaluate whether DP-SPD preserves utility across different generation tasks, model scales, and query distributions, we conduct experiments on WikiText-2, Alpaca, Dolly-15k, CNN/DailyMail, and WildChat using LLaMA-2 and Qwen3 models. We make the following key observations.

DP-SPD Recovers Generation Quality Most Effectively under Tight Privacy. Figure 4 shows that increasing k and r yields the largest gains on WikiText-2 under small privacy budgets. With $r = 1$ and $k = 39$, DP-SPD reaches PPL 14.93 at $\epsilon = 2.0$, close to the plaintext PPL of 14.76; the differences quickly diminish as ϵ increases.

DP-SPD Preserves Higher Utility than the Evaluated Baselines on Instruction Generation. As shown in Figure 3, DP-SPD remains close to plaintext utility on both Alpaca and Dolly-15k, achieving consistently lower PPL and higher ROUGE-L than the evaluated style-adapted private-inference baselines. This advantage holds across both datasets, indicating that DP-SPD better preserves fluency and task-relevant content under privacy protection. These results demonstrate its effectiveness for practical instruction-generation tasks.

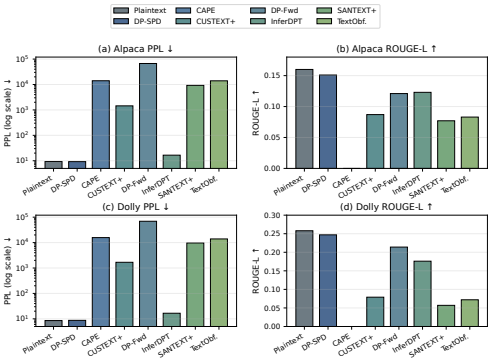


Figure 3: Comparison with related private-inference methods on instruction generation using LLaMA-2-7B-Chat at $\epsilon = 2.5$. Lower PPL and higher ROUGE-L are better; PPL uses a logarithmic scale.

The Advantage Persists on Stronger Models and Real-World Queries. Table 2 shows that DP-SPD scales to stronger models and realistic generation settings. On Qwen3-14B/CNN-DailyMail, it achieves a PPL of 14.11 and ROUGE-L of 0.1555, remaining close to the plaintext results of 12.92 and 0.1572 while outperforming private baselines. On Qwen3-32B/WildChat, DP-SPD exactly matches plaintext utility in QA accuracy and open-ended quality. These results demonstrate that DP-SPD preserves its utility advantage across model scales, datasets, and generation formats.

Table 2: Utility on stronger models and real-world queries. CNN/DailyMail uses Qwen3-14B at $\epsilon = 1.5$; WildChat uses Qwen3-32B at $\epsilon = 3$.

	Qwen3-14B / CNN-DM		Qwen3-32B / WildChat	
Method	**PPL ↓**	**R-L ↑**	**QA Acc. ↑**	**Open ↑**
Plaintext	12.92	0.1572	**72.6%**	**4.716**
DP-SPD	**14.11**	**0.1555**	**72.6%**	**4.716**
TextObf.	16.88	0.1499	27.6%	2.93
DP-Forward	240441.63	0.1314	21.4%	2.71
CUSTEXT+	2606.63	0.0352	43.8%	3.79
SANTEXT+	18081.09	0.0500	39.6%	3.56
InferDPT	36.37	0.0831	36.6%	3.21

5.3 Privacy Evaluation

We evaluate DP-SPD against three privacy attacks. Query Identification Attack (QIA) tests whether the real query can be distinguished from the shuffled candidates using likelihood-based signals; Embedding Inversion Attack (EIA) first attempts to recover discrete content from the perturbed representations before identifying the real candidate; and the LLM-assisted attack uses GPT-4o-mini to perform semantic candidate selection and query reconstruction.

Prefix-Conditioned Dummies Substantially Reduce Query Identification Risk. Table 3 shows that DP-SPD achieves the lowest QIA ASR and substantially smaller real–dummy PPL gaps on both IMDb and WikiText-2. This confirms that conditioning dummy suffixes on the shared prefix makes them substantially harder to distinguish from the real continuation.

Table 3: QIA and dummy-strategy ablation. Results are averaged over $\epsilon \in \{1.5, 2.0, 2.5, 3.0\}$, $r \in \{20\%, 50\%, 80\%\}$, and $k \in \{9, 19, 29\}$. Lower values are better; bold marks the best ASR and PPL gap.

Dataset	Dummy strategy	ASR (%) ↓	Real PPL	Dummy PPL	PPL gap ↓
IMDb	DP-SPD prefix-consistent	**13.53**	29.32	36.69	**7.38**
	Random-sampled	36.25	29.31	49.32	20.00
	Random-token	97.94	29.31	1282.39	1253.08
WikiText-2	DP-SPD prefix-consistent	**11.89**	129.61	108.92	**31.05**
	Random-sampled	28.83	145.49	440.03	294.54
	Random-token	74.86	153.44	1936.43	1782.99

DP-SPD Remains Robust under Embedding Inversion. As shown in Table 4, increasing k reduces the group-level EIA ASR for all dummy construction strategies. DP-SPD achieves the lowest ASR under both tested dummy numbers. This suggests that DP-SPD also provides stronger candidate-level concealment under embedding inversion.

Larger Anonymity Sets Also Suppress Semantic Attacks. Although GPT-4o-mini is stronger than likelihood-based QIA, Table 5 shows that increasing k reduces selection accuracy, exact match, and real-suffix token accuracy. Thus, larger anonymity sets make candidate identification and private-content reconstruction difficult for semantic adversaries.

Table 4: Group-wise EIA results on IMDb, averaged over $\epsilon \in \{1.5, 3.0\}$ and $r \in \{20\%, 50\%, 80\%\}$. Lower ASR is better.

Dummy strategy	$k = 9$	$k = 19$	$k = 29$
DP-SPD prefix-cond.	**8.00**	**4.50**	**2.50**
Random-sampled	9.50	5.33	2.67
Random-token	8.83	5.33	3.00

API-Only Utility Is Robust to Client Representations. Changing the client embedding space alters token retention by 20.65 percentage points but GPT-4o accuracy by only 0.8 points. This small task-level difference indicates that DP-SPD is not strongly tied to a specific client embedding space and remains effective and reliable in API-only deployment settings.

Table 5: GPT-based attack results. Lower values indicate stronger privacy protection.

Metric	$k = 9$	$k = 19$	$k = 29$
Selection Acc. (%) ↓	22.67	13.33	**10.33**
Exact Match (%) ↓	9.25	6.83	**5.33**
Full-Query Token Acc. (%) ↓	50.09	**48.50**	48.76
Real-Suffix Token Acc. (%) ↓	18.35	11.41	**10.07**

5.4 Efficiency Evaluation

A practical deployment of DP-SPD introduces two additional sources of overhead: client-side dummy generation and server-side inference over the augmented candidate set. We therefore evaluate both the cost of the continuation model and the benefit of shared-prefix KV-cache reuse.

Continuation-model scale exposes a clear quality–cost trade-off. Figure 5 studies the client-side continuation model while keeping the privacy configuration ($\epsilon = 3$, $k = 9$, $r = 0.2$) and evaluation setting fixed. Larger continuation models generally generate more plausible dummy suffixes: Surface ROUGE-L increases from 0.1138 with Qwen2.5-0.5B to 0.1691 with Qwen2.5-32B, while PPL decreases accordingly. This improvement, however, comes with higher client-side generation cost; the 32B model requires 3.756 s per candidate group compared with 0.813 s for the 0.5B model. Importantly, QIA success does not increase monotonically with model size, suggesting that improving dummy quality does not necessarily make the real query easier to identify.

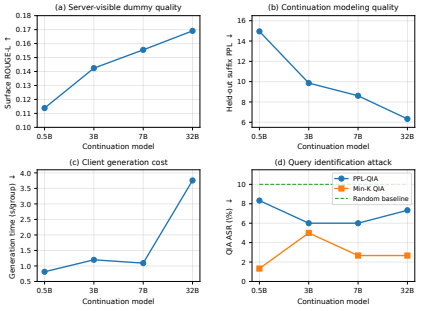


Figure 5: Effect of continuation-model scale on dummy quality, held-out suffix PPL, generation time, and QIA ASR. Higher Surface ROUGE-L is better; lower values are better for all other metrics.

Shared-prefix KV caching substantially reduces inference overhead. Let $p = 1 - r$ denote the fraction of tokens in the reusable shared prefix. Without prefix sharing, every real and dummy candidate must be processed independently, increasing inference time with the candidate-set size. DP-SPD instead computes the shared prefix once and reuses its KV cache across all suffix branches. As shown in Figure 6, for 1024-token sequences with $k = 24$, shared-prefix reuse improves inference efficiency from 3868.02 ms to 1002.86 ms at $p = 0.8$, corresponding to a $3.86\times$ speedup. The benefit becomes more pronounced as the anonymity set grows, with speedup increasing from $2.87\times$ at $k = 9$ to $3.86\times$ at $k = 24$.

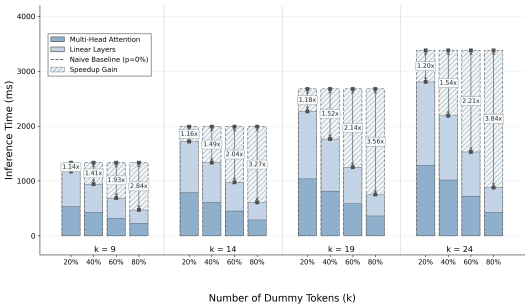


Figure 6: Inference efficiency with shared-prefix KV-cache reuse for 1024-token inputs. Larger shared-prefix ratios and larger anonymity sets yield greater computational savings.

This result also reveals an important practical trade-off. Increasing the dummy ratio r allows a larger portion of the query to benefit from dummy-based shuffling, but shortens the reusable prefix and therefore increases server-side computation. Conversely, smaller r preserves a longer shared prefix and provides substantial efficiency gains. Thus, r and k jointly control not only privacy and utility, but also inference efficiency, allowing DP-SPD to select practical operating points without processing every dummy as an independent full sequence.

6 Conclusion

We presented DP-SPD, a shuffle-based privacy-preserving inference framework for LLMs. The framework achieves a favorable balance among privacy, utility, and efficiency. Experimental results on classification, generation, and attack-based privacy evaluation show that DP-SPD can substantially reduce inference-time privacy leakage while maintaining utility close to plaintext inference. Our current framework adopts a **fixed prefix–suffix split**, without explicitly identifying where sensitive information appears in the query. A promising direction is to first detect privacy-sensitive spans and then restructure the query accordingly, for example by moving sensitive content toward positions where shuffle-based protection is more effective, potentially yielding a better privacy–utility trade-off. Another limitation is that our current analysis focuses on single-query inference rather than **multi-turn conversations** or **long-context** settings. In such scenarios, repeated occurrences of the same token or semantic content may be perturbed independently into inconsistent outputs, which can accumulate utility degradation across turns or over long contexts. Extending DP-SPD to preserve consistency under repeated queries and long-context composition is therefore an important direction for future work.

Acknowledgment

We thank the anonymous reviewers for their insightful comments, which helped improve the quality of this work. We also gratefully acknowledge the support received for this research. This research is supported in part by the National Natural Science Foundation of China (Grant No. 62441238), the National Natural Science Foundation of China under Grant U2441240 (“Ye Qisun” Science Foundation), and the Xiaomi Young Scholars’ Key Research Program via Open Competition.

References

[1] T. B. Brown, B. Mann, N. Ryder *et al.*, “Language models are few-shot learners,” *NeurIPS*, 2020.

[2] L. Ouyang, J. Wu, X. Jiang *et al.*, “Training language models to follow instructions with human feedback,” *NeurIPS*, 2022.

[3] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *Journal of machine learning research*, vol. 24, no. 240, pp. 1–113, 2023.

[4] A. C. Qwen Team, "Qwen2 technical report," https://qwen2.org/paper/, 2025.

[5] K. Chen, X. Zhou, Y. Lin, S. Feng, L. Shen, and P. Wu, "A survey on privacy risks and protection in large language models," *Journal of King Saud University Computer and Information Sciences*, vol. 37, no. 7, p. 163, 2025.

[6] T. Bai, Y. Feng, and S. Fu, "Safeguarding user data privacy in online large language model services," *Journal of Systems Architecture*, p. 103555, 2025.

[7] D. Rathee, D. Li, I. Stoica, H. Zhang, and R. Popa, "Mpc-minimized secure llm inference," *arXiv preprint arXiv:2408.03561*, 2024.

[8] W. Zeng, Y. Dong, J. Zhou, J. Tan, L. Wang, T. Wei, R. Wang, and M. Li, "Mpcache: Mpc-friendly kv cache eviction for efficient private llm inference," *Advances in Neural Information Processing Systems*, vol. 38, pp. 97 254–97 284, 2026.

[9] L. de Castro, A. Polychroniadou, and D. Escudero, "Privacy-preserving large language model inference via gpu-accelerated fully homomorphic encryption," in *Neurips Safe Generative AI Workshop 2024*, 2024.

[10] R. Zhang, Z. Zheng, and W. Bao, "Practical secure inference algorithm for a fine-tuned large language model based on fully homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 21, pp. 17–29, 2025.

[11] M. Du, X. Yue, S. S. Chow, T. Wang, C. Huang, and H. Sun, "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*, 2023, pp. 2665–2679.

[12] P. Mai, R. Yan, Z. Huang, Y. Yang, and Y. Pang, "Split-and-denoise: Protect large language model inference with local differential privacy," in *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024, pp. 34 281–34 302.

[13] M. Tong, K. Chen, J. Zhang, Y. Qi, W. Zhang, N. Yu, T. Zhang, and Z. Zhang, "Inferdpt: Privacy-preserving inference for black-box large language models," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[14] Y. Lin, R. Yang, Y. Mao, Q. Zhang, J. Hong, Q. Cai, Y. Wu, H. Liu, Z. Chen, B. Duan *et al.*, "Obfuslm: Privacy-preserving language model service against embedding inversion attacks," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025, pp. 1160–1174.

[15] R. Thareja, P. Nakov, P. Vepakomma, and N. Lukas, "Dp-fusion: Token-level differentially private inference for large language models," in *International Conference on Learning Representations*, vol. 2026, 2026, pp. 151 088–151 120.

[16] Z. Liu, Y. Cheng, S. Liu, R. Wen, Y. Hu, and Z. Qin, "Dp-mind: Differentially private memory-guided long-context inference," *arXiv preprint arXiv:2610.13132*, 2026.

[17] Y. Hu, J. Lou, J. Liu, W. Ni, F. Lin, Z. Qin, and K. Ren, "ERASER: Machine unlearning in MLaaS via an inference serving-aware approach," in *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '24. ACM, 2024, pp. 3883–3897. [Online]. Available: https://doi.org/10.1145/3658644.3670398

[18] X. Zhang, H. Xu, Z. Ba, Z. Wang, Y. Hong, J. Liu, Z. Qin, and K. Ren, "Privacyasst: Safeguarding user privacy in tool-using large language model agents," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 6, pp. 5242–5258, 2024.

[19] T. Dong, Y. Meng, S. Li, G. Chen, Z. Liu, and H. Zhu, "Depth gives a false sense of privacy:{LLM} internal states inversion," in *34th USENIX Security Symposium (USENIX Security 25)*, 2025, pp. 1629–1648.

[20] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology-EUROCRYPT 2006*. Springer, 2006, pp. 486–503.

[21] X. Wu, F. Li, A. Kumar, K. Chaudhuri, S. Jha, and J. Naughton, "Bolt-on differential privacy for scalable stochastic gradient descent-based analytics," in *Proceedings of the 2017 ACM international conference on management of data*, 2017, pp. 1307–1322.

[22] A. Bittau, U. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, and B. Seefeld, "Prochlo: Strong privacy for analytics in the crowd," in *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, 2017, pp. 441–459.

[23] Ú. Erlingsson, V. Feldman, I. Mironov, A. Raghunathan, K. Talwar, and A. Thakurta, "Amplification by shuffling: From local to central differential privacy via anonymity," in *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2019, pp. 2468–2479.

[24] X. Li, W. Liu, H. Feng, K. Huang, Y. Hu, J. Liu, K. Ren, and Z. Qin, "Privacy enhancement via dummy points in the shuffle model," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 3, pp. 1001–1016, 2024.

[25] R. Pope, S. Douglas, A. Chowdhery, J. Devlin, J. Bradbury, J. Heek, K. Xiao, S. Agrawal, and J. Dean, "Efficiently scaling transformer inference," *Proceedings of Machine Learning and Systems*, vol. 5, pp. 606–624, 2023.

[26] Z. Luo, S. Shao, S. Zhang, L. Zhou, Y. Hu, C. Zhao, Z. Liu, and Z. Qin, "Shadow in the cache: Unveiling and mitigating privacy risks of KV-cache in LLM inference," in *Proceedings of the 33rd Annual Network and Distributed System Security Symposium (NDSS 2026)*. San Diego, CA, USA: The Internet Society, Feb. 2026. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/shadow-in-the-cache-unveiling-and-mitigating-privacy-risks-of-kv-cache-in-llm-inference/

[27] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi, "Broadening the scope of differential privacy using metrics," in *Privacy Enhancing Technologies*. Springer, 2013, pp. 82–102.

[28] O. Feyisetan, B. Balle, T. Drake, and T. Diethe, "Privacy- and utility-preserving textual analysis via calibrated multivariate perturbations," in *Proceedings of the Thirteenth ACM International Conference on Web Search and Data Mining (WSDM)*. ACM, 2020, pp. 178–186.

[29] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Foundations and Trends in Theoretical Computer Science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[30] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, "Llama 2: Open foundation and fine-tuned chat models," *arXiv preprint arXiv:2307.09288*, 2023.

[31] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*. Association for Computational Linguistics, 2011, pp. 142–150.

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "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)*, 2019, pp. 4171–4186.

[33] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," in *Proceedings of the 2018 EMNLP workshop BlackboxNLP: Analyzing and interpreting neural networks for NLP*, 2018, pp. 353–355.

[34] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," in *International Conference on Learning Representations (ICLR)*, 2017.

[35] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford alpaca: An instruction-following llama model," 2023.

[36] M. Conover, M. Hayes, A. Mathur, J. Xie, J. Wan, S. Shah, A. Ghodsi, P. Wendell, M. Zaharia, and R. Xin, "Free dolly: Introducing the world's first truly open instruction-tuned llm," 2023.

[37] A. Yang *et al.*, "Qwen3 technical report," 2025.

[38] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1073–1083.

[39] K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in Neural Information Processing Systems*, vol. 28, 2015, pp. 1693–1701.

[40] F. Jelinek, R. L. Mercer, L. R. Bahl, and J. M. Baker, "Perplexity—a measure of the difficulty of speech recognition tasks," *Journal of the Acoustical Society of America*, vol. 62, 1977.

[41] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*. Association for Computational Linguistics, 2004, pp. 74–81.

[42] W. Zhao, X. Ren, J. Hessel, C. Cardie, Y. Choi, and Y. Deng, "Wildchat: 1m chatGPT interaction logs in the wild," in *The Twelfth International Conference on Learning Representations*, 2024.

[43] Q. Team, "Qwen2.5 technical report," 2025. [Online]. Available: https://arxiv.org/abs/2412.15115

[44] C. Song and A. Raghunathan, "Information leakage in embedding models," in *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, 2020, pp. 377–390.

[45] OpenAI, "Gpt-4o mini: advancing cost-efficient intelligence," https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/, 2024, accessed: 2026-05-04.

[46] R. Plant, D. Gkatzia, and V. Giuffrida, "Cape: Context-aware private embeddings for private language learning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 7970–7978.

[47] X. Zhou, Y. Lu, R. Ma, T. Gui, Y. Wang, Y. Ding, Y. Zhang, Q. Zhang, and X.-J. Huang, "Textobfuscator: Making pre-trained language model a privacy protector via obfuscating word representations," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 5459–5473.

[48] X. Yue, M. Du, T. Wang, Y. Li, H. Sun, and S. S. Chow, "Differential privacy for text analytics via natural text sanitization," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 3853–3866.

[49] S. Chen, F. Mo, Y. Wang, C. Chen, J.-Y. Nie, C. Wang, and J. Cui, "A customized text sanitization mechanism with differential privacy," in *Findings of the Association for Computational Linguistics: ACL 2023*, 2023, pp. 5747–5758.

[50] V. Feldman, A. McMillan, and K. Talwar, "Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling," in *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2021, pp. 954–964.

[51] P. Kairouz, S. Oh, and P. Viswanath, "The composition theorem for differential privacy," in *International conference on machine learning*. PMLR, 2015, pp. 1376–1385.

A Algorithm Details

This section provides the detailed procedures used in DP-SPD. Algorithm 2 perturbs the prefix embeddings and projects them back to tokens to obtain the privatized prefix. Algorithm 3 constructs the real and dummy candidate sequences by reusing the noisy prefix embeddings and independently perturbing their suffix embeddings. Algorithm 4 specifies the nearest-neighbor projection used to convert noisy embedding sequences into tokens, both individually and in batches.

Algorithm 2 PRIVATIZEPREFIX

Require: Prefix $\mathcal{P} = [x_1, \ldots, x_T]$, embedding function E, projection operator $\Pi_{\mathcal{V}}$, privacy budget ϵ_{pre}

Ensure: Noisy prefix embeddings $\widetilde{\mathcal{P}}_{\text{emb}}$ and privatized token prefix $\widehat{\mathcal{P}}$

Initialize $\widetilde{\mathcal{P}}_{\text{emb}} \leftarrow [\,]$
for $t = 1$ to T **do**
  Sample $\mathbf{z}_t \sim \text{Lap}(1/\epsilon_{\text{pre}})$
  Append $E(x_t) + \mathbf{z}_t$ to $\widetilde{\mathcal{P}}_{\text{emb}}$
end for
$\widehat{\mathcal{P}} \leftarrow$ PROJECTTOTOKENS$(\widetilde{\mathcal{P}}_{\text{emb}}, E, \Pi_{\mathcal{V}})$
return $(\widetilde{\mathcal{P}}_{\text{emb}}, \widehat{\mathcal{P}})$

Algorithm 3 BUILDCANDIDATES

Require: Noisy prefix embeddings $\widetilde{\mathcal{P}}_{\text{emb}}$, real suffix $\mathcal{S}$, dummy suffixes $\{\mathcal{D}^{(i)}\}_{i=1}^{k}$, embedding function E, privacy budget ϵ_{suf}

Ensure: Noisy candidate sequences $H^{(0)}, \ldots, H^{(k)}$

for $i = 0$ to k **do**
  Initialize $H^{(i)} \leftarrow \widetilde{\mathcal{P}}_{\text{emb}}$
end for
for $t = 1$ to $|\mathcal{S}|$ **do**
  Sample $\mathbf{z}_{0,t} \sim \text{Lap}(1/\epsilon_{\text{suf}})$
  Append $E(\mathcal{S}[t]) + \mathbf{z}_{0,t}$ to $H^{(0)}$
  for $i = 1$ to k **do**
    Sample $\mathbf{z}_{i,t} \sim \text{Lap}(1/\epsilon_{\text{suf}})$
    Append $E(\mathcal{D}^{(i)}[t]) + \mathbf{z}_{i,t}$ to $H^{(i)}$
  end for
end for
return $H^{(0)}, \ldots, H^{(k)}$

Algorithm 4 PROJECTTOTOKENS and PROJECTBATCH

Require: Noisy embedding sequence $Z = [\mathbf{h}_1, \ldots, \mathbf{h}_L]$, embedding function E, projection operator $\Pi_{\mathcal{V}}$

Ensure: Projected token sequence $\widehat{X}$

$\widehat{X} \leftarrow [\,]$
for $\ell = 1$ to L **do**
  $\widehat{x}_\ell \leftarrow \Pi_{\mathcal{V}}(\mathbf{h}_\ell) = \arg\min_{w \in \mathcal{V}} \|\mathbf{h}_\ell - E(w)\|_2$
  Append $\widehat{x}_\ell$ to $\widehat{X}$
end for
return $\widehat{X}$
For a batch: PROJECTBATCH$([Z^{(0)}, \ldots, Z^{(k)}], E, \Pi_{\mathcal{V}})$
  return [PROJECTTOTOKENS$(Z^{(0)}, E, \Pi_{\mathcal{V}}), \ldots,$ PROJECTTOTOKENS$(Z^{(k)}, E, \Pi_{\mathcal{V}})$]

B A Natural-Language Example

To provide an intuitive illustration of DP-SPD, we present an end-to-end example adapted from a real user query in the WildChat dataset [42]. The example demonstrates query partitioning, prefix-conditioned dummy generation, shuffling, server-side inference, and client-side answer recovery.

Original user query:
what is a biggest salary for Embedded Developer in Automotive industry with 7 year experience

Shared prefix $\mathcal{P}$:
what is a biggest salary for Embedded

Private suffix $\mathcal{S}$:
Developer in Automotive industry with 7 year experience

Shuffled candidates ($k = 4$):

Candidate 0: *Dummy*
what is a biggest salary for Embedded Systems Architect with at least 5 years of experience in Germany

Candidate 1: *Real*
what is a biggest salary for Embedded Developer in Automotive industry with 7 year experience

Candidate 2: *Dummy*
what is a biggest salary for Embedded Systems Engineer in Bangalore, India

Candidate 3: *Dummy*
what is a biggest salary for Embedded Firmware Engineer in China

Candidate 4: *Dummy*
what is a biggest salary for Embedded Systems Programmer with more than 3 years of experience in Europe

Privacy perturbation and dummy generation. The client first perturbs the shared prefix $\mathcal{P}$ to obtain its privatized representation and the projected prefix $\widehat{\mathcal{P}}$. A lightweight continuation model then generates four dummy suffixes conditioned on $\widehat{\mathcal{P}}$. The real suffix and dummy suffixes are independently perturbed before transmission. For readability, the example displays the underlying natural-language candidates rather than their noisy representations; the projected prefix and suffix tokens may differ after perturbation. The real/dummy labels are provided only for illustration and are not revealed to the server.

Shuffling and server-side inference. Let $H^{(0)}$ denote the privatized real candidate and $H^{(1)}, \ldots, H^{(4)}$ denote the privatized dummy candidates. In this example, the client applies the permutation

$$\pi = (1, 0, 2, 3, 4),$$

producing the shuffled batch

$$H = [H^{(1)}, H^{(0)}, H^{(2)}, H^{(3)}, H^{(4)}].$$

The server performs inference on the shuffled, privatized candidates and returns the corresponding responses in the same order. Depending on the deployment mode, the server receives either noisy embeddings or their projected token sequences, rather than the raw queries shown above.

Client-side answer de-shuffling. The client records the real candidate's position as

$$j = \pi^{-1}(0) = 1.$$

After receiving the ordered response batch

$$\mathcal{Y} = [y^{(1)}, y^{(0)}, y^{(2)}, y^{(3)}, y^{(4)}],$$

the client retrieves

$$y_{\text{real}} = \mathcal{Y}[1] = y^{(0)},$$

where zero-based indexing is used. Thus, the client recovers the response corresponding to the privatized real query without revealing its position to the server.

C Theoretical Proofs

Proof of Theorem 4.2. Fix a privatized prefix $\widehat{\mathcal{P}}$. Consider two possible real suffixes under the same prefix, and let $\boldsymbol{s}$ and $\boldsymbol{s}'$ denote their embedding representations. Define their suffix distance as

$$c = \|\boldsymbol{s} - \boldsymbol{s}'\|.$$

Let $\boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k$ be the dummy suffix embeddings generated from the same privatized prefix $\widehat{\mathcal{P}}$. Since the dummy suffix generator uses only $\widehat{\mathcal{P}}$ and is independent of the real suffix, replacing $\boldsymbol{s}$ with $\boldsymbol{s}'$ changes only the real suffix embedding. We denote the two corresponding suffix candidate groups by

$$\mathcal{S} = \{\boldsymbol{s}, \boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k\}, \qquad \mathcal{S}' = \{\boldsymbol{s}', \boldsymbol{d}_1, \boldsymbol{d}_2, \ldots, \boldsymbol{d}_k\}.$$

Following Lemma 3.3 in [50], there exists a randomized post-processing algorithm f such that the canonical shuffled output distributions can be written as

$$P_0 = f(A+1, C-A), \qquad Q_0 = f(A, C-A+1),$$

where $C \sim \mathrm{Bin}(k, p)$ and $A \sim \mathrm{Bin}(C, 1/2)$. According to Lemma A.1 in [50], if

$$p \geq \frac{16\ln(2/\delta)}{k+1},$$

then P_0 and Q_0 are (η, δ)-indistinguishable, where

$$\eta = \log\left(1 + \frac{8\sqrt{\log(4/\delta)}}{\sqrt{p(k+1)}} + \frac{8}{p(k+1)}\right).$$

Since the local randomizer on suffix embeddings satisfies ϵ-d_χ-privacy, it satisfies standard pure DP with privacy budget $c\epsilon$ on the pair $(\boldsymbol{s}, \boldsymbol{s}')$. By Lemma 3.4 in [51], there exists a randomized algorithm

$$\mathcal{Q} : \{0,1\} \to \mathcal{Y}$$

such that

$$\mathcal{R}(\boldsymbol{s}) = \frac{e^{c\epsilon}}{e^{c\epsilon}+1}\mathcal{Q}(0) + \frac{1}{e^{c\epsilon}+1}\mathcal{Q}(1),$$

and

$$\mathcal{R}(\boldsymbol{s}') = \frac{1}{e^{c\epsilon}+1}\mathcal{Q}(0) + \frac{e^{c\epsilon}}{e^{c\epsilon}+1}\mathcal{Q}(1).$$

Therefore, the shuffled outputs of the two suffix candidate groups can be expressed as

$$\mathcal{M}_{DP\text{-}SPD}(\mathcal{S}) \overset{d}{=} P = f(A+\Delta, C-A+1-\Delta),$$

and

$$\mathcal{M}_{DP\text{-}SPD}(\mathcal{S}') \overset{d}{=} Q = f(A+1-\Delta, C-A+\Delta),$$

where

$$\Delta \sim \mathrm{Bern}\left(\frac{e^{c\epsilon}}{e^{c\epsilon}+1}\right).$$

Let

$$U = \frac{1}{2}(P_0 + Q_0).$$

Then

$$P = \frac{2}{e^{c\epsilon}+1}U + \frac{e^{c\epsilon}-1}{e^{c\epsilon}+1}P_0,$$

and

$$Q = \frac{2}{e^{c\epsilon}+1}U + \frac{e^{c\epsilon}-1}{e^{c\epsilon}+1}Q_0.$$

By the hockey-stick divergence argument,

$$\max\left\{D_{e^{\eta'}}(U\|P_0), D_{e^{\eta'}}(Q_0\|U)\right\} \leq D_{e^{\eta'}}(P_0\|Q_0).$$

Thus, P and Q are $(\eta', p\delta)$-indistinguishable, where

$$\eta' \leq \log\left(1 + \frac{e^{c\epsilon} - 1}{e^{c\epsilon} + 1}\left(\frac{8\sqrt{\log(4/\delta)}}{\sqrt{p(k+1)}} + \frac{8}{p(k+1)}\right)\right).$$

Since $p \in (0, 1]$, we have $p\delta \leq \delta$. Therefore, for any measurable output set $\mathcal{A}$,

$$\Pr[\mathcal{M}_{DP\text{-}SPD}(\mathcal{S}) \in \mathcal{A}] \leq e^{\eta'} \Pr[\mathcal{M}_{DP\text{-}SPD}(\mathcal{S}') \in \mathcal{A}] + \delta.$$

Taking $p = e^{-\epsilon}$ gives

$$\eta' \leq \log\left(1 + \frac{e^{c\epsilon} - 1}{e^{c\epsilon} + 1}\left(\frac{8\sqrt{e^{\epsilon}\log(4/\delta)}}{\sqrt{k+1}} + \frac{8e^{\epsilon}}{k+1}\right)\right).$$

Writing $\eta' = \epsilon'' c$, we obtain

$$\epsilon'' = \frac{1}{c}\log\left(1 + \frac{e^{c\epsilon} - 1}{e^{c\epsilon} + 1}\left(\frac{8\sqrt{e^{\epsilon}\log(4/\delta)}}{\sqrt{k+1}} + \frac{8e^{\epsilon}}{k+1}\right)\right),$$

which matches the bound stated in Theorem 4.2. Hence, DP-SPD satisfies the stated ϵ''-d_χ privacy bound for the suffix part with probability at least $1 - \delta$. □

Proof of Theorem 4.4. We consider the two possible cases.

Case 1: The queries differ in the prefix.. In this case, $\mathcal{S} = \mathcal{S}'$. Thus, $F(\cdot, \mathcal{S})$ is the same randomized post-processing for both queries. By Theorem 4.1 and the post-processing property,

$$\begin{aligned}\Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}) \in \mathcal{A}] &\leq e^{\epsilon_{\text{pre}} d_\chi(\mathcal{P}, \mathcal{P}')} \Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}') \in \mathcal{A}] \\ &= e^{\epsilon_{\text{pre}} d_\chi(\mathcal{Q}, \mathcal{Q}')} \Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}') \in \mathcal{A}].\end{aligned}$$

Case 2: The queries differ in the suffix.. In this case, $\mathcal{P} = \mathcal{P}'$, so the privatized prefix follows the same distribution $\mu_{\mathcal{P}}$ for both queries. For every fixed privatized prefix p, Theorem 4.2 gives

$$\Pr[F(p, \mathcal{S}) \in \mathcal{A}] \leq e^{\epsilon_{\text{suf}} d_\chi(\mathcal{S}, \mathcal{S}')} \Pr[F(p, \mathcal{S}') \in \mathcal{A}] + \delta.$$

Integrating over p and using $\int \mu_{\mathcal{P}}(dp) = 1$, we obtain

$$\begin{aligned}\Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}) \in \mathcal{A}] &\leq e^{\epsilon_{\text{suf}} d_\chi(\mathcal{S}, \mathcal{S}')} \Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}') \in \mathcal{A}] + \delta \\ &= e^{\epsilon_{\text{suf}} d_\chi(\mathcal{Q}, \mathcal{Q}')} \Pr[\mathcal{M}_{\text{DP-SPD}}(\mathcal{Q}') \in \mathcal{A}] + \delta.\end{aligned}$$

Therefore, both cases satisfy the claimed guarantee with $\epsilon_{\text{qry}} = \max\{\epsilon_{\text{pre}}, \epsilon_{\text{suf}}\}$. □

D Hardware Configuration

All experiments were conducted on 4 NVIDIA RTX A6000 GPUs with 48GB memory. Utility experiments with LLaMA-2-7B and Qwen3-14B required approximately 30 and 50 GPU-hours, respectively. Attack-based privacy evaluations required approximately 30 GPU-hours in total. Efficiency experiments were measured on the same hardware with batch size and sequence length specified in Section 5.4.

E Additional Experimental Results

This appendix provides the complete experimental results that are omitted from the main paper for space. We organize the material following the same three axes as the main text: utility, privacy, and efficiency. Unless otherwise stated, r denotes the dummy ratio and $p = 1 - r$ denotes the shared-prefix ratio. For methods whose original pipelines are not directly compatible with our LLaMA/Qwen evaluation stack, we report *style-adapted* implementations under a unified model, dataset, and evaluator harness; the exact adaptations are summarized in Table 18.

Table 6: API-only token projection diagnostics at $\epsilon = 3$. Token recovery is the fraction of active tokens unchanged after nearest-token projection.

Dataset	Samples	Active tokens	Changed	Recovery
SST-2	872	20,199	18	99.9109%
QNLI	5,463	262,051	323	99.8767%

Table 7: Projection-induced interface gap on SST-2. Results are mean $\pm$ standard deviation over five noise seeds.

Input / Interface	Accuracy (%)	Δ vs. DP Emb.
Clean embedding	94.20	–
Clean API-only	94.20	–
DP-SPD embedding	89.13 ± 0.24	–
DP-SPD API-only projection	87.46 ± 0.31	-1.67 pp

E.1 Utility Evaluation Details

E.1.1 API-only Projection Diagnostics

Table 6 reports the token-level/API-only implementation used in the main classification comparison. The target privacy budget is $\epsilon = 3$, with $r = 0.6$, $\epsilon_{\text{private}} = 3.07$, $\epsilon_{\text{non-sensitive}} = 1.8$, clipping norm 1, and seed 42. Special tokens are kept unchanged; all other active tokens are obtained by nearest-neighbor projection of the privatized embedding to the vocabulary of the same checkpoint.

The resulting API-only accuracies are 90.60% on SST-2 and 91.31% on QNLI, compared with 91.06% and 90.36% for the embedding-level DP-SPD setting. Since each token-level result uses one fixed noise seed, the QNLI difference should be interpreted as run-to-run variation rather than an improvement caused by projection.

To isolate the interface effect, we additionally compare direct DP embedding inference and nearest-token projection on 500 SST-2 examples using five independent noise seeds at $\epsilon = 1.5$. The clean embedding and clean token paths are identical, while projection introduces only a 1.67 percentage-point additional loss under DP perturbation (Table 7).

E.1.2 Classification Results

We retain the complete IMDb sweeps from the previous version. Table 8 varies the dummy ratio at $k = 29$, while Table 9 isolates the effect of increasing the anonymity-set size. These results show that the benefit of larger k is strongest under tighter privacy budgets.

Table 8: Classification Accuracy on IMDb using LLaMA-2-7B ($k = 29$).

Method	Dummy Ratio	Privacy Budget (ϵ) 1.5	2.0	2.5	3.0	4.0
Plaintext	-			0.820		
d_χ-privacy	-	0.663	0.758	0.812	0.812	0.817
DP-SPD	$r = 0.1$	0.672	0.782	0.810	0.813	0.817
	$r = 0.2$	0.689	0.797	0.809	0.813	0.815
	$r = 0.4$	0.726	0.809	0.813	0.816	0.815
	$r = 0.6$	0.762	0.811	0.813	0.817	0.816
	$r = 0.8$	0.796	0.812	0.814	0.818	0.817
	$r = 1.0$	0.812	0.818	0.819	0.819	0.820

Table 9: The impact of the number of dummy inputs (k) on LLaMA-2-7B model accuracy under different privacy budgets (ϵ) using IMDb dataset. The results demonstrate that utility generally improves as k increases.

Method	**Dummy Ratio**	ϵ	$k = 9$	$k = 19$	$k = 29$	$k = 39$
Plaintext	-	-		0.820		
DP-SPD	0.1	1.5	0.659	0.666	0.672	0.672
		1.8	0.724	0.732	0.737	0.741
		2.0	0.779	0.781	0.782	0.782
DP-SPD	0.6	1.5	0.664	0.714	0.762	0.784
		1.8	0.788	0.806	0.808	0.807
		2.0	0.810	0.812	0.811	0.814
DP-SPD	1.0	1.5	0.658	0.777	0.811	0.813
		1.8	0.810	0.816	0.814	0.815
		2.0	0.814	0.816	0.818	0.816

Table 10: IMDb classification accuracy (%) under the unified LLaMA-2-7B evaluation harness. DP-SPD uses the fixed configuration $r = 1, k = 29$; the remaining private baselines are style-adapted implementations.

Method	$\epsilon = 1.5$	$\epsilon = 2.0$	$\epsilon = 2.5$	$\epsilon = 3.0$	$\epsilon = 4.0$
Clean control	81.85	81.85	81.85	81.85	81.85
DP-SPD ($r = 1, k = 29$)	**81.20**	**81.80**	**81.90**	**81.90**	**82.00**
CAPE-style	50.98	50.68	50.92	51.17	51.72
CUSTEXT-style	59.62	59.38	59.68	60.68	63.45
DP-Forward-style	50.28	50.30	50.13	50.16	50.28
InferDPT-style	50.40	50.49	52.43	52.88	54.46
SANTEXT-style	50.39	50.66	50.38	50.49	50.49
TextObfuscator-style	49.96	50.27	51.18	53.32	56.27

E.1.3 Generation Results on WikiText-2, Alpaca, and Dolly

The following tables retain the complete DP-SPD parameter sweeps from the previous version and add unified style-adapter comparisons. For the cross-method comparison, DP-SPD is evaluated at a fixed configuration for every privacy budget rather than selecting the best r or k separately at each ϵ.

Table 11: Utility comparison (PPL and ROUGE-L) on **WikiText-2**. We compare Baselines and DP-SPD with varying Dummy Ratios (r). At $\epsilon = 1.5$, utility is generally poor due to extreme noise. At $\epsilon = 2.0$, increasing r significantly improves utility by reducing the impact of context noise. At $\epsilon \geq 2.5$, DP-SPD converges to the plaintext baseline regardless of r.

Method	**Metric**	$\epsilon = 1.5$					$\epsilon = 2.0$					$\epsilon = 2.5$					$\epsilon = 3.0$				
		$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 49$	$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 49$	$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 49$	$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 49$
Plaintext	PPL	14.76																			
	ROUGE	0.194																			
d_{χ}-privacy	PPL	747.61					63.14					20.59					16.31				
	ROUGE	0.037					0.129					0.174					0.193				
DP-SPD with varying Dummy Ratio (r)																					
$r = 0.1$	PPL	734.12	570.66	525.02	490.48	484.47	48.00	42.64	41.58	41.12	40.59	19.01	18.92	18.91	18.89	18.85	16.09	16.07	16.06	16.07	16.07
	ROUGE	0.040	0.044	0.050	0.054	0.054	0.138	0.146	0.147	0.142	0.146	0.181	0.181	0.183	0.185	0.181	0.190	0.192	0.192	0.192	0.191
$r = 0.2$	PPL	719.85	423.16	344.25	305.93	284.33	36.76	31.58	30.33	29.88	29.44	18.03	17.95	17.85	17.80	17.80	15.83	15.82	15.78	15.80	15.78
	ROUGE	0.037	0.054	0.069	0.072	0.079	0.154	0.153	0.157	0.153	0.155	0.180	0.185	0.184	0.182	0.180	0.187	0.189	0.185	0.186	0.182
$r = 0.4$	PPL	692.66	244.47	156.92	122.27	106.90	25.79	22.59	21.89	21.56	21.41	16.79	16.62	16.54	16.55	16.57	15.41	15.39	15.39	15.39	15.36
	ROUGE	0.044	0.075	0.098	0.115	0.119	0.167	0.174	0.174	0.177	0.179	0.186	0.188	0.183	0.185	0.189	0.190	0.194	0.194	0.196	0.193
$r = 0.6$	PPL	740.58	144.90	72.82	52.68	45.51	20.85	18.70	18.29	18.20	18.02	15.95	15.81	15.76	15.76	15.77	15.22	15.19	15.18	15.14	15.14
	ROUGE	0.043	0.096	0.131	0.142	0.143	0.176	0.181	0.186	0.182	0.187	0.191	0.192	0.189	0.192	0.191	0.195	0.192	0.193	0.200	0.196
$r = 0.8$	PPL	735.70	82.00	36.53	27.42	24.51	17.95	16.52	16.31	16.20	16.09	15.40	15.23	15.22	15.18	15.16	14.97	14.95	14.92	14.90	14.92
	ROUGE	0.039	0.115	0.150	0.167	0.170	0.185	0.184	0.192	0.189	0.190	0.193	0.188	0.193	0.193	0.193	0.195	0.192	0.195	0.199	0.195
$r = 1.0$	PPL	745.59	48.96	21.52	17.54	16.47	16.23	15.21	15.01	14.93	14.89	14.92	14.82	14.79	14.78	14.77	14.79	14.76	14.75	14.74	14.75
	ROUGE	0.036	0.133	0.172	0.190	0.196	0.194	0.193	0.194	0.192	0.194	0.189	0.196	0.196	0.199	0.195	0.198	0.198	0.194	0.196	0.193

For reference, the InferDPT-style extractor-side conditional PPLs are 77.32, 77.00, 82.12, and 81.69 for $\epsilon = 1.5, 2.0, 2.5, 3.0$, respectively; these values are reported only as diagnostics and are not used for cross-method PPL ranking.

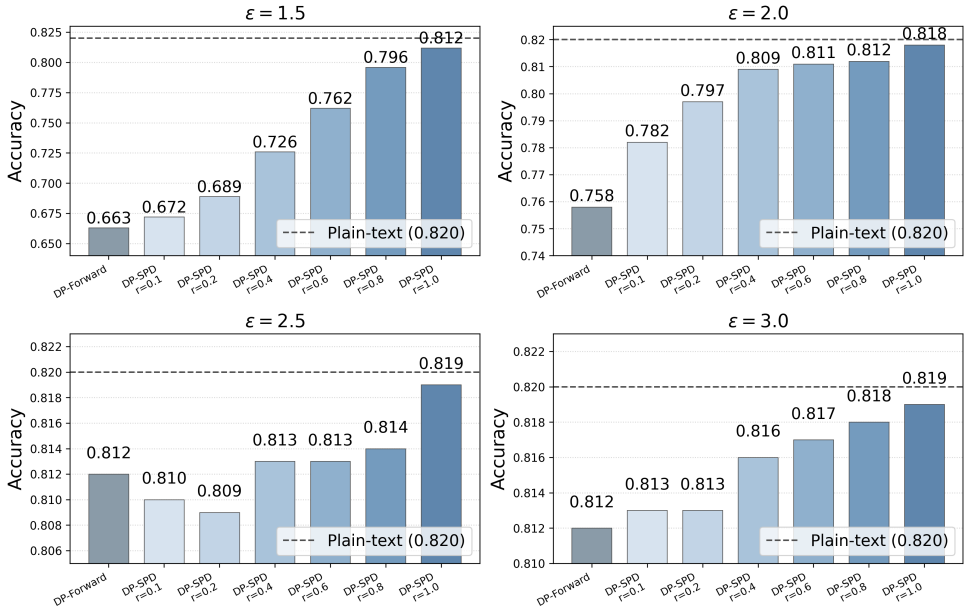


Figure 7: IMDb classification accuracy under different privacy budgets and dummy ratios. This visualization was moved from the main text to the appendix for space.

Table 12: WikiText-2 generation comparison under the unified LLaMA-2-7B harness. Each entry is PPL / ROUGE-L. DP-SPD uses $r = 1, k = 39$.

Method	$\epsilon = 1.5$	$\epsilon = 2.0$	$\epsilon = 2.5$	$\epsilon = 3.0$
Plaintext	14.76 / .194	14.76 / .194	14.76 / .194	14.76 / .194
DP-SPD	**17.54 / .190**	**14.93 / .192**	**14.78 / .199**	**14.74 / .196**
CAPE-style	21853.10 / .000	20549.84 / .000	19633.91 / .000	18942.00 / .000
CUSTEXT-style	3177.39 / .095	2938.88 / .093	2647.31 / .097	2297.97 / .100
DP-Forward-style	87500.65 / .168	78406.94 / .168	68875.98 / .163	60178.53 / .158
SANTEXT-style	10729.42 / .090	10697.31 / .089	10623.72 / .084	10452.40 / .085
TextObfuscator-style	34437.38 / .090	29455.69 / .087	14533.64 / .092	7473.66 / .095

The corresponding InferDPT-style extractor conditional PPLs are 15.89/16.60/16.94 on Alpaca and 16.29/16.62/16.50 on Dolly for $\epsilon = 1.5/2.5/4.0$.

E.1.4 Qwen3-14B on CNN/DailyMail

We retain the complete DP-SPD grid over $\epsilon \in \{1.0, 1.5, 2.0\}$, $r \in \{0.2, 0.4, 0.6, 0.8\}$, and $k \in \{9, 19, 29, 39\}$, followed by the unified related-work comparison.

The InferDPT-style extractor conditional PPLs are 36.29, 36.37, and 36.28 for $\epsilon = 1.0, 1.5, 2.0$. At $\epsilon = 1.0$, TextObfuscator-style obtains lower PPL than DP-SPD (25.68 vs. 35.13), while DP-SPD obtains higher ROUGE-L (.1504 vs. .1351); at $\epsilon \in \{1.5, 2.0\}$, DP-SPD obtains both lower comparable PPL and slightly higher ROUGE-L.

E.1.5 Real-world WildChat Utility

We evaluate Qwen3-32B on 1,000 independent WildChat queries: 500 answerable QA requests and 500 open-ended requests, with target $\epsilon = 3$. QA utility is evaluated by semantic correctness against a reference answer, while open-ended responses are scored from 1 to 5 for overall response quality.

The reported overall ROUGE-L is approximately 0.241 for both plaintext and DP-SPD. The QA judge accepts semantic paraphrases and equivalent concise answers; for open-ended prompts, it assigns a 1–5 quality score based primarily on the original user request.

Table 13: Utility comparison (PPL and ROUGE-L) on the cleaned version of **Stanford Alpaca**. We compare Baselines and DP-SPD with varying Dummy Ratios (r). Consistent with WikiText-2 results, utility is low at $\epsilon = 1.5$ but improves significantly with higher r and larger ϵ. DP-SPD shows clear benefits from context preservation at lower ϵ.

Method	**Metric**	$\epsilon = 1.5$				$\epsilon = 2.5$				$\epsilon = 4.0$			
		$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 9$	$k = 19$	$k = 29$	$k = 39$
Plaintext	PPL	9.34											
	ROUGE	0.160											
d_χ-privacy	PPL	431.93				14.43				9.56			
	ROUGE	0.059				0.134				0.144			
DP-SPD with varying Dummy Ratio (r)													
$r = 0.2$	PPL	433.55	256.42	211.78	191.67	11.90	11.60	11.55	11.49	9.48	9.53	9.48	9.51
	ROUGE	0.050	0.071	0.072	0.075	0.131	0.130	0.137	0.133	0.145	0.145	0.149	0.147
$r = 0.4$	PPL	446.27	156.40	106.99	83.95	10.86	10.55	10.45	10.43	9.43	9.44	9.45	9.45
	ROUGE	0.047	0.094	0.100	0.109	0.137	0.136	0.136	0.144	0.146	0.147	0.150	0.149
$r = 1.0$	PPL	427.13	34.28	15.06	11.87	9.37	9.24	9.25	9.25	9.47	9.48	9.48	9.48
	ROUGE	0.057	0.128	0.133	0.132	0.144	0.144	0.145	0.151	0.155	0.151	0.152	0.153

Table 14: Utility comparison (PPL and ROUGE-L) on **Databricks-Dolly-15k**. The trends are consistent with other datasets: DP-SPD effectively balances privacy and utility, particularly at moderate noise levels ($\epsilon = 2.5$). At high privacy ($\epsilon = 1.5$), DP-SPD with higher r shows improved text quality (ROUGE).

Method	**Metric**	$\epsilon = 1.5$				$\epsilon = 2.5$				$\epsilon = 4.0$			
		$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 9$	$k = 19$	$k = 29$	$k = 39$
Plaintext	PPL	8.55											
	ROUGE	0.258											
d_χ-privacy	PPL	409.17				13.62				9.31			
	ROUGE	0.064				0.234				0.243			
DP-SPD with varying Dummy Ratio (r)													
$r = 0.2$	PPL	382.84	228.19	178.00	160.03	11.41	11.24	11.18	11.07	9.11	9.06	9.05	9.06
	ROUGE	0.063	0.093	0.116	0.114	0.234	0.245	0.244	0.250	0.240	0.248	0.246	0.243
$r = 0.4$	PPL	393.74	144.08	89.58	65.81	10.47	10.26	10.23	10.17	8.89	8.92	8.88	8.89
	ROUGE	0.075	0.096	0.133	0.161	0.241	0.237	0.239	0.236	0.257	0.249	0.252	0.246
$r = 1.0$	PPL	410.38	33.29	14.34	11.31	9.10	8.86	8.80	8.73	8.61	8.63	8.63	8.61
	ROUGE	0.062	0.181	0.233	0.240	0.241	0.253	0.245	0.247	0.259	0.253	0.249	0.253

E.1.6 Style-adapter Implementation Details

Tables 18 shows the details.

E.2 Privacy Evaluation Details

E.2.1 Query Identification Attack (QIA)

We use the term *Query Identification Attack (QIA)* for the attack previously referred to as membership inference in earlier drafts. The adversary observes the shuffled set of $k + 1$ candidates and attempts to identify the real query. Tables 19 and 20 provide the full dummy-strategy breakdown over $r \in \{20\%, 50\%, 80\%\}$ and $k \in \{9, 19, 29\}$, averaged over $\epsilon \in \{1.5, 2.0, 2.5, 3.0\}$.

E.2.2 Embedding Inversion Attack

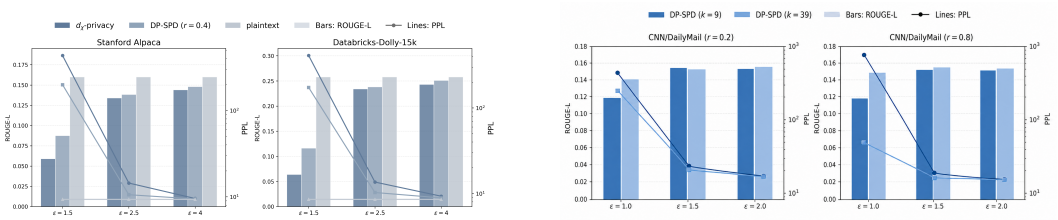
For EIA, the adversary first maps each privatized embedding to its nearest vocabulary token and then performs group-level candidate identification. Results are averaged over $\epsilon \in \{1.5, 3.0\}$ and $r \in \{20\%, 50\%, 80\%\}$. Increasing the anonymity-set size reduces the group-level ASR for all dummy strategies.

E.2.3 LLM-assisted Semantic Identification and Reconstruction

The LLM-assisted attack asks GPT-4o-mini both to select the most plausible real candidate and to reconstruct the private query. Table 5 reports the complete benchmark attack metrics averaged over the tested privacy budgets and dummy ratios.

Table 15: Instruction-generation comparison under the unified LLaMA-2-7B harness. Each entry is PPL / ROUGE-L. DP-SPD uses $r = 1, k = 39$.

Dataset	Method	$\epsilon = 1.5$	$\epsilon = 2.5$	$\epsilon = 4.0$
Alpaca	Plaintext	9.34 / .160	9.34 / .160	9.34 / .160
	DP-SPD	**11.87 / .132**	**9.25 / .151**	**9.48 / .153**
	CAPE-style	17191.15 / .000	14053.85 / .000	12754.43 / .000
	CUSTEXT-style	1856.30 / .085	1445.68 / .087	832.20 / .092
	DP-Forward-style	86864.22 / .116	67077.54 / .121	48869.19 / .121
	SANTEXT-style	9374.88 / .078	9253.24 / .077	9031.51 / .076
	TextObfuscator-style	30357.85 / .082	13963.50 / .083	2732.50 / .093
Dolly	Plaintext	8.55 / .258	8.55 / .258	8.55 / .258
	DP-SPD	**11.31 / .240**	**8.73 / .247**	**8.61 / .253**
	CAPE-style	18498.71 / .000	15722.25 / .000	14218.52 / .000
	CUSTEXT-style	2050.00 / .080	1679.10 / .079	950.35 / .091
	DP-Forward-style	88334.92 / .219	69046.99 / .214	49849.64 / .215
	SANTEXT-style	9796.14 / .062	9652.38 / .057	9482.65 / .060
	TextObfuscator-style	30970.81 / .075	13930.29 / .072	2953.12 / .070



(a) LLaMA-2-7B on Alpaca and Dolly-15k.

(b) Qwen3-14B on CNN/DailyMail.

Figure 8: Generation-utility visualizations moved from the main text to the appendix.

Table 16: Generation performance of Qwen3-14B on CNN/DailyMail summarization. We report PPL and ROUGE-L under different privacy budgets, dummy ratios, and dummy numbers. Lower PPL and higher ROUGE-L are better.

ϵ	r	PPL ↓				ROUGE-L ↑			
		$k = 9$	$k = 19$	$k = 29$	$k = 39$	$k = 9$	$k = 19$	$k = 29$	$k = 39$
1.0	0.2	253.50	161.76	141.97	137.18	0.1197	0.1361	0.1391	0.1411
1.0	0.4	315.52	128.56	97.74	90.92	0.1173	0.1387	0.1388	0.1403
1.0	0.6	403.63	98.93	64.93	58.43	0.1209	0.1396	0.1433	0.1469
1.0	0.8	553.70	71.02	40.13	35.13	0.1184	0.1431	0.1507	0.1504
1.5	0.2	16.07	16.04	16.05	16.05	0.1545	0.1541	0.1526	0.1531
1.5	0.4	15.51	15.49	15.48	15.48	0.1533	0.1511	0.1514	0.1513
1.5	0.6	14.90	14.86	14.85	14.84	0.1507	0.1497	0.1524	0.1503
1.5	0.8	14.18	14.12	14.11	14.11	0.1526	0.1550	0.1548	0.1555
2.0	0.2	13.33	13.33	13.33	13.33	0.1535	0.1551	0.1542	0.1556
2.0	0.4	13.28	13.28	13.28	13.28	0.1533	0.1545	0.1535	0.1549
2.0	0.6	13.21	13.21	13.21	13.21	0.1555	0.1552	0.1546	0.1543
2.0	0.8	13.13	13.13	13.13	13.12	0.1526	0.1538	0.1540	0.1544

E.2.4 WildChat Attack Evaluation

We further evaluate 400 shuffled WildChat candidate groups with likelihood-based QIA and GPT-4o-mini semantic attacks. The candidate-selection random baseline is $1/(k+1)$, i.e., 10% for $k = 9$ and 5% for $k = 19$. This baseline applies to candidate selection and should not be interpreted as the baseline for exact-query reconstruction.

The likelihood-based attacks fall below the random-selection level in these experiments, whereas GPT-based semantic selection retains a measurable advantage over random guessing.

E.2.5 Representation Robustness

To test whether API-only behavior is tied to a particular client embedding space, we keep GPT-4o fixed as the downstream backend on MMLU-Pro and change only the client-side projection model.

Table 17: CNN/DailyMail generation comparison with Qwen3-14B. Each entry is PPL / ROUGE-L. DP-SPD uses $r = 0.8, k = 39$.

Method	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = 2.0$
DP-SPD	**35.13 / .1504**	**14.11 / .1555**	**13.12 / .1544**
CAPE-style	118680.03 / .0030	207624.46 / .0028	200676.06 / .0045
CUSTEXT-style	2897.53 / .0288	2606.63 / .0352	2297.36 / .0400
DP-Forward-style	285789.84 / .1264	240441.63 / .1314	228022.08 / .1318
SANTEXT-style	18150.13 / .0524	18081.09 / .0500	18139.80 / .0553
TextObfuscator-style	25.68 / .1351	16.88 / .1499	14.65 / .1533

Table 18: Style-adapted baselines used in the unified LLaMA/Qwen evaluation harness. These are controlled adaptations of the core privacy mechanisms rather than exact reproductions of every original training pipeline.

Adapter	**Preserved core mechanism**	**Adaptation / omitted component**
CAPE-style	Min–max normalization followed by Laplace perturbation	The original private-attribute adversary and task-head training are not reproduced in the unified harness.
CUSTEXT-style	Top-k exponential token mechanism	Original word-vector reconstruction is replaced by the LLaMA/Qwen BPE vocabulary and the common evaluator.
DP-Forward-style	Hidden-state perturbation	Full noisy fine-tuning from the original pipeline is not reproduced for the unified classification/generation evaluator.
InferDPT-style	RANTEXT-style perturbation and two-stage data flow	IMDb probing does not reproduce the complete remote-LLM/local-extractor pipeline; generation PPL uses an extractor-side conditional definition and is therefore not directly ranked against server-side PPL.
SANTEXT-style	Exponential token sanitization	The mechanism is adapted to the tokenizer/BPE vocabulary of the current target model and the unified evaluator.
TextObfuscator-style	Client-boundary hidden-state perturbation	Full joint training, prototype losses, and other task-specific training components are not reproduced.

Table 19: Detailed QIA results on IMDb. Results are averaged over $\epsilon \in \{1.5, 2.0, 2.5, 3.0\}$. Lower ASR and lower PPL gap are better.

Dummy Strategy	r	**QIA ASR (%)** $\downarrow$			**PPL Gap** $\downarrow$		
		$k = 9$	$k = 19$	$k = 29$	$k = 9$	$k = 19$	$k = 29$
DP-SPD	20%	26.50	22.50	21.00	7.41	7.38	7.42
DP-SPD	50%	13.50	10.25	8.00	8.39	8.16	8.11
DP-SPD	80%	7.75	6.00	6.25	6.89	6.08	6.53
Random-Sampled	20%	54.75	48.00	39.50	19.40	19.59	19.96
Random-Sampled	50%	42.00	31.75	31.25	22.14	21.64	21.03
Random-Sampled	80%	32.25	25.75	21.00	18.09	19.92	18.25
Random-Token	20%	95.75	95.75	93.50	90.64	89.96	90.44
Random-Token	50%	99.25	99.75	99.00	629.69	621.41	627.56
Random-Token	80%	99.75	99.00	99.75	3114.79	3010.73	3002.47

Although the two embedding spaces yield very different token-retention rates, downstream task accuracy differs by only 0.8 percentage points.

Table 20: Detailed QIA results on WikiText-2. Results are averaged over $\epsilon \in \{1.5, 2.0, 2.5, 3.0\}$. Lower ASR and lower PPL gap are better.

Dummy Strategy	r	QIA ASR (%) ↓			PPL Gap ↓		
		$k=9$	$k=19$	$k=29$	$k=9$	$k=19$	$k=29$
DP-SPD	20%	24.00	17.75	13.50	16.26	15.49	14.89
DP-SPD	50%	18.00	13.75	12.25	6.86	7.24	6.08
DP-SPD	80%	4.00	1.50	2.25	72.95	69.94	69.72
Random-Sampled	20%	54.75	42.50	31.75	99.46	98.86	96.29
Random-Sampled	50%	36.25	29.25	25.75	278.44	259.27	243.80
Random-Sampled	80%	17.50	13.00	8.75	519.25	542.04	513.41
Random-Token	20%	78.00	74.00	67.50	316.59	307.21	306.04
Random-Token	50%	82.50	78.75	74.50	1490.54	1518.27	1512.71
Random-Token	80%	75.50	72.50	70.50	3480.72	3528.30	3586.50

Table 21: QIA and semantic-reconstruction results on WildChat. Lower values indicate stronger privacy.

Attack Metric	$k=9$	$k=19$	Random selection
Min-K QIA ASR	4.00%	1.75%	10% / 5%
PPL QIA ASR	4.25%	2.50%	10% / 5%
GPT selection ASR	20.0%	13.0%	10% / 5%
GPT exact-query recovery	18.25%	11.0%	–

E.3 Efficiency Evaluation Details

E.3.1 Continuation-model Scale

We vary only the client-side continuation generator among Qwen2.5-0.5B, 3B, 7B, and 32B. All other factors are fixed: WikiText-2 test with 300 samples, target $\epsilon = 3$, $k = 9$, $r = 0.2$, $\epsilon_{\text{non-sensitive}} = 3.0$, $\epsilon_{\text{private}} = 7.82$, Laplace noise with clipping norm 1, Qwen2.5-0.5B-Instruct as the transport/projection model, and Qwen2.5-0.5B-Instruct as the common quality/QIA scorer. Every group contains one real candidate and nine dummies, so random candidate selection is 10%.

Table 23: Effect of continuation-model scale. Surface ROUGE-L is measured on the final DP-perturbed and projected dummy suffix. Held-out PPL is scored by each tested proxy itself (not server-side generated-text PPL). QIA uses a fixed 0.5B scorer. Bracketed values indicate the reported intervals.

(a) Dummy quality and generation time

Proxy	Params	Surface R-L ↑	Held-out PPL ↓	s/group ↓
0.5B	0.494B	.1138 [.1067,.1211]	14.95 [13.83,16.15]	**0.813**
3B	3.086B	.1424 [.1332,.1522]	9.86 [9.16,10.63]	1.198
7B	7.616B	.1555 [.1450,.1663]	8.61 [7.97,9.29]	1.092
32B	32.764B	**.1691 [.1547,.1841]**	**6.33 [5.83,6.89]**	3.756

(b) Throughput and query-identification attacks

Proxy	Total (s)	tok/s ↑	PPL-QIA ↓	Min-K QIA ↓
0.5B	243.93	**321.5**	8.33% [5.71,12.01]	**1.33% [0.52,3.38]**
3B	359.35	218.2	**6.00% [3.83,9.28]**	5.00% [3.05,8.08]
7B	327.75	239.3	**6.00% [3.83,9.28]**	2.67% [1.36,5.17]
32B	1126.69	69.6	7.33% [4.89,10.85]	2.67% [1.36,5.17]

Surface ROUGE-L increases monotonically with proxy size. The fixed 0.5B evaluator PPL on the final dummy candidates is 68.26, 67.58, 71.69, and 73.77 for the 0.5B/3B/7B/32B proxies, respectively, and therefore does not improve monotonically. Generation time measures CUDA-synchronized `proxy_model.generate` only and excludes CPU DP noise and nearest-token projection. The attack point estimates remain around or below the 10% random baseline without a monotonic increase as the proxy becomes larger.

Table 22: Representation-space robustness for API-only DP-SPD on MMLU-Pro. Values are reported as mean ± standard deviation.

Projection Model	Dim.	Token Retention	GPT-4o Acc.
Qwen2.5-0.5B	896	$98.50 \pm 0.11\%$	$54.8 \pm 3.03\%$
Gemma-2-2B	2,304	$77.85 \pm 0.15\%$	$54.0 \pm 2.55\%$
Qwen − Gemma	–	+20.65 pp	+0.8 pp

E.3.2 KV-cache Inference-Time Breakdown

Let $p = 1 - r$ denote the shared-prefix ratio. The no-sharing baseline uses $p = 0$ and processes each candidate independently. Table 24 provides the complete inference-time breakdown for sequence length 1024. The strongest measured operating point, $p = 0.8$ ($r = 0.2$) and $k = 24$, reduces total inference time from 3868.02 ms to 1002.86 ms, corresponding to a $3.86\times$ speedup.

Table 24: Inference-time breakdown for 1024-token inputs. MHA, linear-layer, and total CUDA times are in milliseconds. Speedups use the no-sharing baseline ($p = 0$) at the same dummy count k.

p	k	MHA (ms)	Linear (ms)	Total (ms)	MHA gain	Total gain
0.0 (Naive)	0	67.34	84.35	175.07	1.00×	1.00×
	9	607.90	723.57	1527.27	1.00×	1.00×
	14	899.69	1089.36	2281.81	1.00×	1.00×
	19	1216.77	1460.89	3066.87	1.00×	1.00×
	24	1526.78	1852.62	3868.02	1.00×	1.00×
0.2	0	68.54	86.04	177.48	0.98×	0.99×
	9	533.55	639.53	1346.36	1.14×	1.13×
	14	788.00	929.74	1968.51	1.14×	1.16×
	19	1038.61	1229.86	2596.67	1.17×	1.18×
	24	1285.44	1520.59	3212.09	1.19×	1.20×
0.4	0	68.66	86.15	177.68	0.98×	0.99×
	9	426.36	516.64	1078.04	1.43×	1.42×
	14	609.08	725.42	1527.90	1.48×	1.49×
	19	814.14	949.42	2018.03	1.49×	1.52×
	24	1018.54	1171.71	2500.22	1.50×	1.55×
0.6	0	69.10	86.83	178.84	0.97×	0.98×
	9	319.80	368.66	787.81	1.90×	1.94×
	14	450.98	522.17	1109.93	1.99×	2.06×
	19	585.56	664.19	1425.83	2.08×	2.15×
	24	723.86	805.43	1744.20	2.11×	2.22×
0.8	0	68.68	86.19	177.69	0.98×	0.99×
	9	226.49	243.16	532.95	2.68×	2.87×
	14	290.66	317.05	690.63	3.10×	3.30×
	19	362.52	390.01	855.83	3.36×	3.58×
	24	426.51	453.97	1002.86	3.58×	3.86×

E.3.3 Privacy–Utility–Efficiency Trade-off

The two system parameters k and r jointly determine the effective perturbation level and server-side cost. Increasing k strengthens shuffle amplification, while increasing the dummy ratio r protects a larger portion of the input but shortens the reusable prefix. The complete trade-off is visualized in Figure 9. At comparable noise levels, a small dummy ratio (e.g., $r = 0.2$, hence $p = 0.8$) can achieve more than a 3.2× inference speedup while maintaining similar privacy protection.

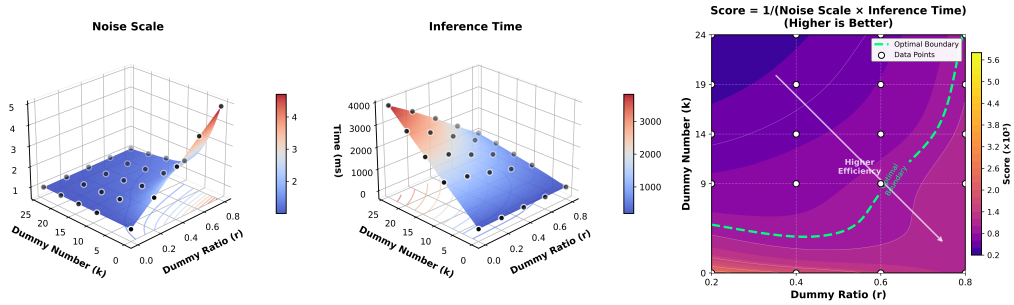


Figure 9: Trade-off between noise scale and inference time across dummy ratios r and dummy numbers k.

The KV-cache efficiency evaluation also shows the same trend at $r = 0.6$: for $k = 39$, inference time decreases from 2919.36 ms without prefix reuse to 1906.68 ms with shared-prefix reuse, a 34.7% reduction, while the associated classification-accuracy change is below 0.25 percentage points in the matched comparison. These results illustrate that KV-cache reuse can offset a substantial fraction of the compute overhead introduced by larger anonymity sets.

F Prompt for GPT Inference Attack

Prompt used for GPT inference attack

```
You are performing a GPT inference attack adapted from InferDPT.

Context:
- A privacy-preserving inference system produced one group of candidate queries.
- Exactly one candidate is the real user query.
- The other candidates are dummy continuations generated from the same shared
↪  prefix.
- The text you see below was reconstructed from noised embeddings, so it may
↪  contain token errors.

Your tasks:
1. Decide which candidate suffix is most likely to come from the real query.
2. Recover the most likely original full query before noise.
3. Briefly explain which semantic or stylistic cues made that candidate more
↪  plausible than the others.

Return valid JSON only in the following schema:
{
  "predicted_real_index": 0,
  "recovered_query": "full recovered query here",
  "confidence": 0.0,
  "brief_reasoning": "short explanation"
}

Shared prefix:
This was a quite brutalFileSystem. There were huge implausibilities, and a silly
↪  script, bad acting, etc.<br /><br />The only reason to watch this movie is
↪  that from-time to

Candidate suffixes:
Candidate 0 suffix:
 time a person mentions "Dyslexia Syndrome

Candidate 1 suffix:
 time you hear the words "The most expensive movie

Candidate 2 suffix:
 time it reminds you how wonderful a film Director can
```

```
Candidate 3 suffix:
 time someone will askwhy I have no internet connection

Candidate 4 suffix:
 time it shows on TV or through internet streaming services

Candidate 5 suffix:
 time an audience might have noticed that it was really

Candidate 6 suffix:
 time they had a little scene which objectives how good

Candidate 7 suffix:
time the narrator of the scene makes the film look

Candidate 8 suffix:
 time some quite impressive sets of breasts were exposed.

Candidate 9 suffix:
 time it makes me laugh because of how ridiculous one
```

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims in the abstract and introduction are supported by the methodology, privacy analysis, and experiments.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in the Conclusion Section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The assumptions for the privacy analysis are stated in Section 4.2, proofs for the correctness theorem and suffix privacy theorem are provided in the main text and Appendix C.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 5.1 and the appendix describe the models, datasets, privacy budgets, dummy ratios, dummy numbers, attack settings, and evaluation metrics needed to reproduce the main experiments.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The anonymized code is available at: `https://anonymous.4open.science/r/code-of-paper-6C04`

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify the models, datasets, evaluation metrics, privacy budgets, dummy ratios, dummy numbers, and attack configurations in Section 5.1; additional details are provided in the appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not report error bars for all experiments due to the high computational cost of large-model inference and attack evaluation. For robustness, we report results across multiple privacy budgets, dummy ratios, dummy numbers, models, and datasets.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The appendix reports the GPU type, memory, number of devices, approximate runtime for each experiment group, and estimated total compute used for the reported experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The study uses public datasets and open-source or publicly accessible models for privacy-preserving inference research. We do not collect private user data or conduct human-subject experiments, and the work is intended to reduce privacy risks in LLM-as-a-Service settings.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

Justification: Our work aims to mitigate privacy risks by improving the privacy–utility–efficiency trade-off of differentially private LLM inference. The expected societal impact is positive, as our methods strengthen privacy guarantees with acceptable computational overhead, facilitating broader adoption of privacy-aware LLM services. Overall, we believe that the benefits of more privacy-preserving LLM inference outweigh potential risks, and we do not foresee significant negative societal consequences arising uniquely from the techniques proposed in this paper.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release new pretrained language models, image generators, scraped datasets, or other high-risk assets. The proposed method is a privacy-preserving inference mechanism evaluated on existing public datasets and models.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We referenced all datasets, models, and software libraries used in the experiment and complied with their terms of use.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [N/A]

Justification: The paper does not introduce or release new datasets, pretrained models, or benchmarks. The method is evaluated using existing public datasets and models.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing, user studies, or experiments with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects, user studies, or newly collected personal data, so IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are central to the method and experiments. We use a lightweight LLM to generate dummy suffixes and evaluate DP-SPD on open LLMs for classification, generation, attack-based privacy evaluation, and efficiency analysis; these usages are described in Sections 5.1.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.