BEYOND LOCAL PERTURBATION: PRIVACY AMPLIFICATION FOR BLACK-BOX LLM INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

For black-box private LLM inference, differentially private prompt privatization provides formal privacy guarantees without modifying the remote LLM, but stronger privacy typically requires heavier perturbation and degrades downstream utility. We introduce a privacy amplification framework that leverages cross-user anonymity and expands the anonymity set through controllable pseudo-prompt expansion, enabling weaker local perturbation under the same end-to-end privacy budget. To account for the resulting amplification, we develop a mechanism-aware accountant that exploits the sequential structure of prompt privatizers through likelihood-ratio moments, avoiding explicit enumeration of the exponentially large sequence output space. The accountant supports both autoregressive and token-replacement privatizers and remains tractable for sequence lengths where distribution-based mechanism-aware accounting becomes costly. Experiments on MMLU and MedicalQA show relative F1 improvements of 8.2%–10.2% over the original privatization mechanisms and substantially tighter privacy amplification than generic shuffle accounting.

1 INTRODUCTION

Large language models (LLMs) are increasingly accessed through cloud-hosted, black-box services, where users submit prompts to a remote provider and receive generated responses. Such prompts may contain sensitive information, creating substantial privacy concerns when the service provider is not fully trusted Thomas et al. (2025); Mai et al. (2024). This setting is particularly challenging because users generally cannot access or modify the architecture, parameters, or inference procedure of proprietary LLMs. Privacy mechanisms that operate entirely on the user input are therefore especially attractive for black-box LLM inference.

Existing black-box-compatible approaches primarily protect prompts before they reach the service provider. One line of work applies prompt sanitization, identifying sensitive information and masking, replacing, or generalizing it before transmission to the service provider Chong et al. (2025); Staab et al. (2025). While these approaches can preserve useful semantics, they often lack a formal end-to-end privacy guarantee, and removing or replacing sensitive content may also suppress information important for downstream inference Yang et al. (2025). Differentially private (DP) prompt privatization provides a principled alternative by locally randomizing the prompt before transmission Feyisetan et al. (2020); Utpala et al. (2023); Thareja et al. (2026). Such mechanisms provide formal privacy guarantees independent of the remote LLM, but stronger privacy typically requires heavier perturbation and thus lower downstream utility.

The trade-off between privacy and utility motivates stronger protection without imposing additional perturbation on individual prompt. Fortunately, commercial LLM services typically handle many concurrent requests within short time windows and may use batching to improve serving efficiency Yu et al. (2022), providing an opportunity to anonymize user requests through shuffling. By obscuring the association between users and requests, shuffling can provide substantially stronger privacy guarantees than local randomization alone Erlingsson et al. (2019); Feldman et al. (2023).

However, applying shuffling to prompt privatization raises two challenges. First, **strong amplification requires a sufficiently large anonymity set.** Relying solely on concurrent requests may provide limited amplification, while collecting larger batches increases latency. Second, **existing shuffle accounting lacks tractable mechanism-specific guarantees for prompt privatization.** Generic

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

bounds depend mainly on the worst-case local privacy budget and batch size, ignoring mechanism-specific privacy loss and often yielding conservative guarantees Erlingsson et al. (2019); Koskela et al. (2023). Exact accounting, in contrast, requires the full shuffled output distribution and is intractable for sequence outputs with exponentially large support Girgis et al. (2021).

To address these challenges, we introduce a privacy amplification framework for black-box LLM inference. Users locally privatize their prompts, which are anonymously shuffled before reaching the provider. To enlarge the anonymity set, we introduce data-independent pseudo prompts processed by the same privatizer and mixed into the batch. To obtain tight yet tractable privacy guarantees, we develop a mechanism-aware accountant that propagates likelihood-ratio moments through shuffling while avoiding exact distributional enumeration. Empirically, our framework improves F1 by 8.2%–10.2% on average over the original privatization mechanisms.

Our contributions are summarized as follows:

- **Privacy strengthening beyond local perturbation.** We extend black-box private LLM inference from purely local protection to a shuffle-based framework that exploits cross-user anonymity, enabling weaker local perturbation under the same end-to-end privacy guarantee.
- **Mechanism-aware shuffle accounting.** We develop a tractable accountant that preserves the moment profile for likelihood ratio of the local privatizer, yielding a tighter bound than generic accounting while avoiding enumeration of the exponentially large sequence-level output space.
- **Controllable batch expansion.** We incorporate data-independent pseudo prompts into the shuffled batch for enlarging the anonymity set without requiring additional genuine users, and characterize the resulting trade-off between privacy, utility, and complexity.

2 Related Work

2.1 Privacy-Preserving LLM Inference

Existing approaches to private LLM inference include cryptographic protocols Yao (1982); Pang et al. (2024); Xu et al. (2025), trusted execution environments Chrapek et al. (2025); Sridhara et al. (2024), and input perturbation methods Du et al. (2023); Zhang et al. (2025); Zheng et al. (2026). Many methods require access to or modifications to the serving infrastructure, limiting their applicability to proprietary black-box LLM services.

For black-box LLM inference, existing model-agnostic methods typically transform the prompt before it is sent to the provider. Prompt sanitization methods identify and sanitize privacy-sensitive content. For example, Casper Chong et al. (2025) detects and replaces sensitive entities, and LLM-based methods rewrite or generalize identifying information while preserving the remaining semantics Staab et al. (2025); Yang et al. (2025). However, such methods may miss contextual privacy leakage or remove information important for downstream inference. A separate line of work provides formal privacy guarantees by applying differential privacy to text. Token-level mechanisms replace individual tokens based on semantic similarity Feyisetan et al. (2020); Yue et al. (2021); Chen et al. (2023); Awon et al. (2025), while autoregressive mechanisms generate privatized text token by token Utpala et al. (2023); Thareja et al. (2026). These methods rely on local perturbation for stronger privacy protection, whereas our work amplifies the privacy through anonymous shuffling.

2.2 Privacy Amplification by Shuffling

Privacy amplification by shuffling strengthens local privacy by removing the association between users and their randomized reports. Generic shuffle analyses derive amplification guarantees primarily from the local privacy budget and the number of shuffled reports Erlingsson et al. (2019). Subsequent work has tightened these generic guarantees through Rényi-DP analysis and improved approximate-DP bounds Girgis et al. (2021); Feldman et al. (2023), and numerical accounting provides sharper privacy estimates for general LDP shufflers Koskela et al. (2023).

Beyond generic accounting, several works exploit additional structure of the local randomizer. Privacy blanket methods characterize common components among output distributions Balle et al. (2019), variation ratio analyses incorporate likelihood ratio and total variation information Wang et al. (2024), and recent decomposition-based methods derive tighter mechanism-aware bounds through

optimized distributional decompositions Su et al. (2026) or finite-n blanket-divergence analysis beyond pure LDP Takagi & Liew (2026). However, applying these approaches to sequence output is computationally challenging, as the required distributional quantities are defined over exponentially large output spaces.

3 PRELIMINARIES

3.1 DIFFERENTIAL PRIVACY

We first introduce differential privacy (DP) Dwork (2006), which quantifies the indistinguishability of a randomized mechanism on neighboring private inputs.

Definition 3.1 (Differential Privacy)**.** Let $\mathcal{M}$ be a randomized mechanism and let $x \sim x'$ denote two neighboring inputs. The mechanism $\mathcal{M}$ satisfies (ϵ, δ)-differential privacy if, for every pair $x \sim x'$ and every measurable event S, we have:

$$\Pr[\mathcal{M}(x) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(x') \in S] + \delta.$$

To characterize (ϵ, δ)-DP, we use the hockey-stick divergence Zhu et al. (2022).

Definition 3.2 (Hockey-stick Divergence)**.** Let P and Q be two probability density functions, and let $\gamma \geq 1$. Their hockey-stick divergence is defined as:

$$H_{\gamma}(P\|Q) = \int \max\{0, P(y) - \gamma Q(y)\} dy.$$

Using hockey-stick divergence, we can provide a characterization of (ϵ, δ)-DP bounds Zhu et al. (2022). Specifically, $\mathcal{M}$ is (ϵ, δ)-DP if and only if, for every neighboring pair $x \sim x'$, it follows that:

$$H_{e^{\epsilon}}\big(\mathcal{M}(x)\|\mathcal{M}(x')\big) \leq \delta.$$

We further introduce Rényi differential privacy (RDP) Mironov (2017).

Definition 3.3 (Rényi Differential Privacy)**.** For $\alpha > 1$, the order-α Rényi divergence between two probability distributions P and Q is defined as

$$D_{\alpha}(P\|Q) = \frac{1}{\alpha - 1} \log \mathbb{E}_{Y \sim Q} \left[\left(\frac{P(Y)}{Q(Y)} \right)^{\alpha} \right].$$

A randomized mechanism $\mathcal{M}$ satisfies (α, ρ)-Rényi differential privacy (RDP) Mironov (2017) if, for every pair of neighboring inputs $x \sim x'$, $D_{\alpha}\big(\mathcal{M}(x)\|\mathcal{M}(x')\big) \leq \rho$.

3.2 DIFFERENTIALLY PRIVATE PROMPT PRIVATIZATION

Let $x = (x_1, \ldots, x_{T_x}) \in \mathcal{X}$ denote a private input prompt, where x_i is the i-th token and T_x is its length. A differentially private prompt privatizer $\mathcal{M}$ maps x to a privatized sequence $y = (y_1, \ldots, y_{T_y}) \in \mathcal{Y}$, whose output distributions satisfy (ϵ, δ)-DP for neighboring prompts. We denote $P_x(y_{1:T_y}) = \Pr[\mathcal{M}(x) = y_{1:T_y}]$. A broad class of prompt privatizers admits the sequential factorization

$$P_x(y_{1:T_y}) = \prod_{t=1}^{T_y} p_{x,t}(y_t \mid y_{<t}), \tag{1}$$

where $p_{x,t}(\cdot \mid y_{<t})$ is the conditional distribution at step t. This formulation covers both token replacement and autoregressive generation. For brevity, we use T to denote the output length below.

Token replacement. A token-wise replacement mechanism independently perturbs each input token through a randomized channel Feyisetan et al. (2020); Chen et al. (2023). Let $K_t(y_t \mid x_t)$ denote the probability of replacing x_t by y_t at position t. It is a special case of equation 1 with $p_{x,t}(y_t \mid y_{<t}) = K_t(y_t \mid x_t)$, and therefore $P_x^{\mathrm{rep}}(y_{1:T}) = \prod_{t=1}^{T} K_t(y_t \mid x_t)$. The channel K_t may depend on token embeddings or other public information.

Autoregressive generation. An autoregressive mechanism generates a privatized prompt token by token Utpala et al. (2023); Majmudar et al. (2022); Thareja et al. (2026), for example using a lightweight local language model. At step t, it samples $Y_t \sim p_{x,t}(\cdot \mid Y_{<t})$, so the resulting sequence distribution follows exactly the factorization in equation 1. Unlike token replacement, the conditional distribution depends jointly on the private input x and the previously generated tokens.

4 PRIVATE INFERENCE ARCHITECTURE

We introduce our privacy-preserving inference architecture with a black-box LLM service provider, where users locally privatize their prompts before outsourcing inference. Our architecture separates local privacy protection from anonymity amplification, where each prompt is first transformed by a local privatizer, and the resulting reports are then anonymously shuffled before being sent to the service provider.

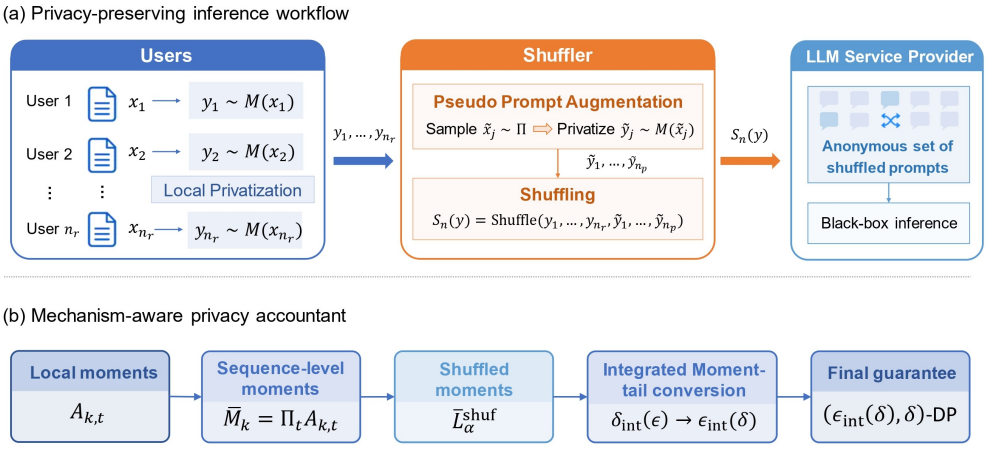


Figure 1: **Overview of the proposed framework.** (a) Private LLM inference with local privatization, pseudo-prompt augmentation, and shuffling. (b) Mechanism-aware privacy accounting from local moments to the final DP guarantee.

4.1 THREAT MODEL AND PRIVACY GOAL

We consider three parties: users, a shuffler, and an LLM service provider. Each user i holds a private prompt x_i and performs all local randomization on a trusted device. The service provider hosts a black-box LLM and is assumed to be honest-but-curious: it follows the prescribed inference procedure but may attempt to infer information about individual users from the received requests. The shuffler is also assumed to follow the protocol and does not collude with the service provider.

The shuffler should not learn the users' prompts, while the service provider should learn neither the raw prompts nor the association between privatized prompts and their originating users. For neighboring datasets X and X' differing in exactly one prompt, our goal is to ensure that the provider's observation satisfies prompt-level (ϵ, δ)-DP. Since subsequent LLM inference is post-processing of the shuffled privatized prompts, the same privacy guarantee extends to the provider's complete inference view.

4.2 LOCALLY PRIVATIZED AND SHUFFLED INFERENCE

For each genuine user $i \in [n_r]$, the private prompt x_i is processed locally by a prompt privatizer $\mathcal{M}$ $y_i \sim \mathcal{M}(x_i)$, and the privatized prompt y_i is transmitted to the shuffler.

The shuffler collects the locally privatized prompts and applies a uniformly random permutation before forwarding them to the service provider. The mechanism is denoted as:

$$\mathcal{S}_{n_r}(X) = \mathsf{Shuf}\big(\mathcal{M}(x_1), \dots, \mathcal{M}(x_{n_r})\big). \tag{2}$$

The service provider then independently performs standard inference on each shuffled report.

This design provides two complementary layers of protection. Local privatization limits the information revealed by each individual report, while shuffling removes the correspondence between reports and users and thereby amplifies the resulting privacy guarantee. Section 5 develops a mechanism-aware accountant that quantifies this amplification using the likelihood-ratio moments of $\mathcal{M}$.

4.3 PSEUDO-PROMPT AUGMENTATION

To strengthen privacy amplification without waiting for additional users, we augment each batch with pseudo prompts, thus providing a controllable way to enlarge the anonymity set.

Let n_p denote the number of pseudo prompts. The shuffler samples $\widetilde{x}_j \overset{\text{i.i.d.}}{\sim} \Pi$ for $j \in [n_p]$ from a public distribution independent of the protected user inputs, and applies the same local privatizer: $\widetilde{y}_j \sim \mathcal{M}(\widetilde{x}_j)$. All genuine and pseudo reports use independent randomization. The shuffler then uniformly permutes the combined batch:

$$S_n(X) = \mathrm{Shuf}\big(\mathcal{M}(x_1), \ldots, \mathcal{M}(x_{n_r}), \mathcal{M}(\widetilde{x}_1), \ldots, \mathcal{M}(\widetilde{x}_{n_p})\big), \quad n = n_r + n_p.$$

The provider processes all requests without genuine or pseudo indicators, and the shuffler discards pseudo responses locally. Increasing n_p can therefore strengthen privacy amplification at the cost of additional generation and inference, providing a tunable trade-off between privacy and complexity.

5 MECHANISM-AWARE SHUFFLING BOUND

We develop a mechanism-aware privacy accountant for shuffled prompt privatization. The key idea is to characterize each local prompt privatizer through its likelihood-ratio moments, propagate the resulting moment profile through shuffling, and then convert the shuffled moments into an (ϵ, δ)-DP guarantee using an integrated moment-tail bound.

5.1 LIMITATIONS OF EXISTING SHUFFLING BOUNDS

Existing shuffling bounds typically characterize privacy amplification using only coarse parameters such as the local privacy budget ϵ and the number of users n. By ignoring the distributional structure of the local privatizer, these worst-case bounds can be overly conservative for a specific mechanism, especially when ϵ is large. In particular, two mechanisms with the same local ϵ receive essentially the same generic guarantee even if their likelihood-ratio distributions differ substantially.

At the other extreme, exact mechanism-specific accounting requires evaluating the full shuffled output distribution. If the local output space has size K, the shuffled output space contains $\binom{n+K-1}{K-1}$ possible histograms. For a length-T prompt over vocabulary $\mathcal{V}$, $K = |\mathcal{V}|^T$, making exact accounting computationally intractable. This motivates our mechanism-aware approach, which retains a compact profile of likelihood-ratio moments rather than reducing the mechanism to a single ϵ or enumerating its full output distribution.

5.2 INTEGRATED MOMENT-TAIL CONVERSION

Let $\widetilde{P}_X$ and $\widetilde{P}_{X'}$ denote the output distributions of the shuffled mechanism $\mathcal{S}_n$ on neighboring datasets $X \sim X'$. We characterize their distinguishability through the moments of the shuffled likelihood ratio. For an integer order $\alpha \geq 2$, let L_α^{shuf} denote the worst-case shuffled log moment, and let $\overline{L}_\alpha^{\mathrm{shuf}}$ be a computable upper bound derived in Section 5.3.

Using these moment bounds, we upper-bound the likelihood-ratio tail and integrate it to obtain a valid hockey-stick divergence bound $\delta_{\mathrm{int}}(\epsilon)$. The resulting privacy guarantee is:

$$\epsilon_{\mathrm{int}}(\delta) = \inf\left\{\epsilon : \delta_{\mathrm{int}}(\epsilon) \leq \delta\right\}.$$

The full moment-tail conversion and its derivation are provided in Appendix B.2.1. Thus, privacy accounting reduces to obtaining a tight shuffled moment profile $\{\overline{L}_\alpha^{\mathrm{shuf}}\}_{\alpha \in \mathcal{A}}$.

5.3 SHUFFLED MOMENTS

Rather than summarizing a local privatizer by a single worst-case privacy budget, we retain its likelihood-ratio moment profile. For neighboring prompts $x \sim x'$, we define:

$$M_k(x, x') = \mathbb{E}_{Y \sim P_{x'}}\left[\left(\frac{P_x(Y)}{P_{x'}(Y)}\right)^k\right],$$

and let $\overline{M}_k$ be a uniform upper bound over neighboring inputs.

Our shuffle analysis combines the local moment profile $\{\overline{M}_k\}_{k\le\alpha}$ with the batch size and a certified overlap condition on the local output distributions to obtain:

$$L_\alpha^{\mathrm{shuf}} \le \overline{L}_\alpha^{\mathrm{shuf}} = \mathsf{ShuffleMoment}_\alpha\big(n, \{\overline{M}_k\}_{k\le\alpha}\big).$$

Unlike generic shuffle accounting, which primarily depends on a single local privacy parameter, this bound preserves substantially more information about the underlying privatizer.

The overlap condition, the closed-form expression for $\mathsf{ShuffleMoment}_\alpha$, and its derivation are given in Appendix B.2.2. The remaining task is therefore to efficiently obtain the local moment bounds $\{\overline{M}_k\}$ for sequence-valued prompt privatizers.

5.4 Mechanism-Aware Local Moments

The main advantage of our accountant is that the sequence-level moments can be bounded without enumerating the exponentially large output space. Using the sequential factorization in equation 1, repeated conditioning gives:

$$M_k(x, x') \le \prod_{t=1}^{T} A_{k,t} = \overline{M}_k,$$

where $A_{k,t}$ is a worst-case conditional moment bound at privatization step t.

The form of $A_{k,t}$ depends on the local privatizer. For token replacement mechanisms, it can be evaluated directly from the replacement channel. For autoregressive mechanisms, direct optimization over all generation histories is generally intractable, and thus we derive $A_{k,t}$ from a bound on the conditional privacy-loss range. These constructions allow the sequence-level moment profile to be computed from per-step quantities rather than by enumerating the sequence output space.

The explicit formulas for token replacement, autoregressive generation, and the bounded-range moment function are provided in Appendix B.2.3.

6 Theoretical Analysis

6.1 Privacy Guarantee

We first establish that the proposed accountant provides an end-to-end privacy guarantee for shuffled prompt privatization.

Theorem 6.1 (Validity of Mechanism-Aware Privacy Accountant)**.** *Suppose n users independently apply the same local privatizer $\mathcal{M}$, whose output distributions $\{P_x\}_{x\in\mathcal{X}}$ have common support and global overlap $\gamma > 0$ as defined in equation 5. Assume that $\mathcal{M}$ follows the sequential factorization in equation 1.*

Then Algorithm 2 computes valid local moment bounds $\{\overline{M}_k\}$ and shuffled log-moment bounds satisfying $L_\alpha^{\mathrm{shuf}} \le \overline{L}_\alpha^{\mathrm{shuf}}$, for $\alpha = 2, \dots, \alpha_{\max}$. Consequently, the returned $\epsilon_{\mathrm{int}}(\delta)$ guarantees that $\mathcal{S}_n$ satisfies $\big(\epsilon_{\mathrm{int}}(\delta), \delta\big)$-DP.

The same guarantee extends to pseudo-prompt augmentation, provided that pseudo prompts are generated independently of the private inputs and processed by the same privatizer.

Corollary 6.2 (Privacy with Pseudo Prompts)**.** *Suppose n_p pseudo prompts are sampled independently of the private prompts from a distribution Π supported on $\mathcal{X}$, and all true and pseudo prompts are independently processed by the same local privatizer $\mathcal{M}$ before shuffling. Under the conditions of Theorem 6.1, Algorithm 2 remains valid with $n = n_r + n_p$.*

6.2 Complexity Analysis of Privacy Accountant

Let $A = \alpha_{\max}$ and C_{loc} denote the cost of evaluating one per-step local moment bound. Computing the local moment profile costs $O(ATC_{\mathrm{loc}})$. The shuffled moment profile can then be computed in

$O(nA^2)$ time using truncated polynomial recurrences up to degree A. The final integrated moment-tail conversion only incurs a one-dimensional search over ϵ.

Hence, the overall complexity of the accountant is $O\big(ATC_{\mathrm{loc}} + nA^2\big)$, up to the negligible final conversion cost. Importantly, the complexity is polynomial in the number of users and prompt length, and avoids enumerating the exponentially large sequence-level output space $|\mathcal{V}|^T$.

6.3 Tightness Analysis

We characterize the tightness of the proposed accountant at several key stages. For a fixed neighboring pair (x, x'), define $r(y) = P_x(y)/P_{x'}(y)$ and $S_m = m^{-1}\sum_{i=1}^m r(Y_i)$ with $Y_i \overset{\text{i.i.d.}}{\sim} P_{x'}$. Given the exact local moments $\{M_k(x, x')\}_{k\le\alpha}$, it holds that:

$$\mathbb{E}[S_m^\alpha] = \frac{\alpha!}{m^\alpha}[z^\alpha]\left(\sum_{k=0}^{\alpha}\frac{M_k(x,x')}{k!}z^k\right)^m,$$

so the propagation from local moments to the conditional shuffled moment is exact. For fixed $\alpha \ge 2$ and $\gamma > 0$, this further gives:

$$\overline{L}_\alpha^{\mathrm{shuf}} = \binom{\alpha}{2}\frac{M_2(x,x')-1}{\gamma n} + O(n^{-2}),$$

showing that the accountant achieves the $O(1/n)$ shuffle-amplification rate.

For autoregressive privatization, $A_{k,t}^{\mathrm{gen}} \le \psi_k(B_t)$ is tight given only the privacy-loss range B_t, since the worst case is attained by a two-point distribution at the range endpoints. In addition, $\delta_{\mathrm{int}}(\epsilon)$ takes the pointwise minimum over all available moment-based tail bounds, and is therefore no worse than using any single Rényi order. The remaining looseness comes mainly from replacing pair-specific moments by uniform bounds $\{\overline{M}_k\}$ and from the global overlap coefficient γ.

7 Experiment

7.1 Experiment Setting

Our method is tested on two datasets: (i) Medical-QA with medical problems Chen et al. (2024); and (ii) MMLU covering diverse knowledge domains Hendrycks et al. (2021). Unless otherwise specified, each batch contains $n_r = 5000$ genuine requests and $n_p = 5000$ pseudo requests. Privacy accounting uses integer moment orders $\mathcal{A} = \{2, \dots, 100\}$ with target $\delta = 10^{-4}$.

7.2 Utility Analysis

We integrate our framework with three DP prompt privatization mechanisms: (i) DP-Prompt Utpala et al. (2023); Mattern et al. (2022) that performs differentially private paraphrase generation through temperature-controlled sampling; (ii) DP-Fusion Thareja et al. (2026) that bounds the influence of sensitive tokens by blending the private output distribution with a public baseline distribution; and (iii) CusText Chen et al. (2023) that provides relaxed DP guarantees by constructing a customized replacement set for each input token. The first two mechanisms instantiate autoregressive privatization, whereas CusText represents token replacement.

Table 1 compares the downstream utility under the same server-side privacy budget $\epsilon = 5$. By exploiting privacy amplification from shuffling, our framework permits weaker local perturbation and therefore consistently improves utility. The F1 score improves by an average of 10.16%, 8.97%, and 8.27% over the original DP-Prompt, DP-Fusion, and CusText mechanisms, respectively.

We compare the trade-off between privacy and utility of the original mechanisms and their integration with our framework in Figure 2. Our framework consistently improves utility across privacy budgets, yielding average F1 gains of 10.9% for DP-Fusion and 8.3% for CusText. Moreover, as the privacy budget increases from $\epsilon = 1$ to 20, the amplified mechanisms improve F1 by an average of 12.7%, compared with 4.4% for the original mechanisms, indicating faster convergence toward non-private utility.

Table 1: **Utility comparison in F1 score.** The same server-side privacy budget is set to $\epsilon = 5$. *Original* applies each DP mechanism without shuffling amplification, while *Amplified* uses our framework. Bold numbers indicate the better result within each mechanism, and green numbers show the improvement of *Amplified* over *Original*.

		MMLU		MedicalQA	
Mechanism	Setting	GPT-5-mini	Gemini-3.1-flash	GPT-5-mini	Gemini-3.1-flash
DP-Prompt	Original	0.603±0.03	0.556±0.04	0.418±0.02	0.378±0.06
	Amplified	**0.654**±0.04 (+0.051)	**0.612**±0.02 (+0.056)	**0.454**±0.02 (+0.036)	**0.429**±0.03 (+0.051)
DP-Fusion	Original	0.662±0.01	0.626±0.02	0.514±0.05	0.414±0.05
	Amplified	**0.698**±0.02 (+0.036)	**0.664**±0.03 (+0.038)	**0.561**±0.03 (+0.047)	**0.477**±0.03 (+0.063)
CusText	Original	0.752±0.03	0.732±0.05	0.627±0.02	0.504±0.05
	Amplified	**0.802**±0.02 (+0.050)	**0.789**±0.05 (+0.057)	**0.683**±0.04 (+0.056)	**0.553**±0.04 (+0.049)

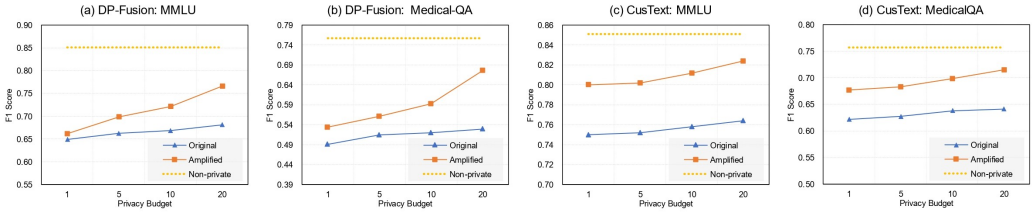


Figure 2: **Utility in F1 score under varying privacy budgets.** We test on MMLU and Medical-QA with DP-Fusion and CusText using GPT-5-mini.

7.3 Complexity Analysis

We first study the trade-off between utility and complexity under a fixed privacy budget. We measure complexity by the batch expansion factor κ, defined as the ratio of the total number of prompts (genuine and pseudo) to the number of genuine requests. A larger κ therefore corresponds to more pseudo prompts used to obfuscate genuine requests. The left two panels of Figure 3 show this trade-off with $n_r = 5000$. As κ increases, the larger shuffled batch provides greater privacy amplification. This allows less local perturbation while maintaining the same end-to-end privacy guarantee, thereby improving utility. Increasing κ from 1 to 20 improves the F1 score by an average of 12.4% for DP-Fusion and 7.7% for CusText.

The right two panels of Figure 3 compare our mechanism-aware privacy bound. The generic accountant provides substantial amplification only when the local privacy budget is small, i.e., $\epsilon \leq 5$, and approaches the local privacy bound when $\epsilon \geq 15$. In contrast, our accountant maintains strong amplification in the regime $\epsilon \geq 15$, achieving amplification factors exceeding $2\times$ and $3\times$ for $\kappa = 2$ and $\kappa = 5$, respectively.

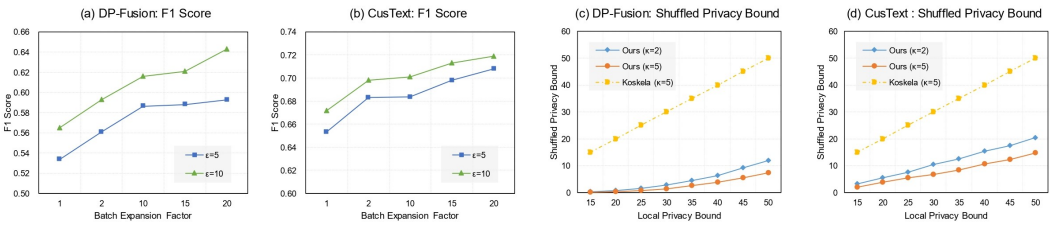


Figure 3: **Utility and shuffled privacy bounds.** (a)–(b): F1 score under varying batch expansion factors on Medical-QA with $n_r = 5000$ using GPT-5-mini. (c)–(d): Shuffled privacy bounds from the generic Koskela accountant and our mechanism-aware accountant.

7.4 Comparison with Mechanism-Aware Accountants

We compare our accountant with two recent mechanism-aware shuffle accountants Su et al. (2026); Takagi & Liew (2026). Both approaches require sequence-level distributional information over up to V^T outputs, causing their cost to grow exponentially with sequence length T.

As shown in Figure 4(a), the computation time of Su et al. (2026) and Takagi & Liew (2026) increases rapidly with T and becomes impractical in our implementation for larger values of T. In contrast, our accountant exploits the autoregressive factorization of DP-Prompt and propagates per-step moment bounds for likelihood ratio without constructing the full V^T output distribution, resulting in much milder growth in runtime as T increases.

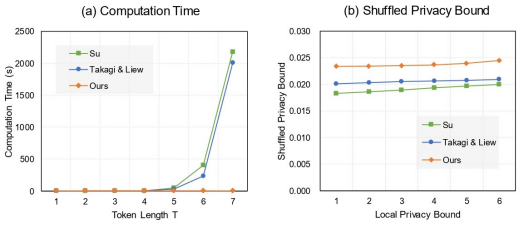


Figure 4: **Comparison with existing mechanism-aware shuffle accountants on DP-Prompt.** (a) Accounting time under varying privatized sequence length T. (b) Shuffled privacy bound under varying local privacy budget with $T = 5$.

Figure 4(b) compares the resulting privacy guarantees at $T = 5$. Su et al. (2026) and Takagi & Liew (2026) achieve tighter bounds by retaining more detailed sequence-level distributional information, whereas our method incurs a modest loss in tightness from using tractable per-step moment envelopes. Nevertheless, it achieves comparable mechanism-aware amplification while avoiding explicit enumeration of the exponentially large sequence space.

7.5 Ablation Study

We evaluate the contribution of shuffling and pseudo-prompt augmentation to downstream utility in Figure 5. *Original* applies the local privatization mechanism without shuffle amplification; *Ours w/o Pseudo* applies our mechanism-aware shuffle accounting using only genuine requests; and *Ours w/ Pseudo* further augments the shuffled batch with pseudo prompts. Across DP-Fusion and CusText on both MMLU and MedicalQA, shuffling alone improves the F1 score by over 3% on average compared with the original setting. Adding pseudo prompts provides a further improvement of over 4% on average, demonstrating that both cross-user shuffling and pseudo-prompt augmentation contribute to the overall utility gain.

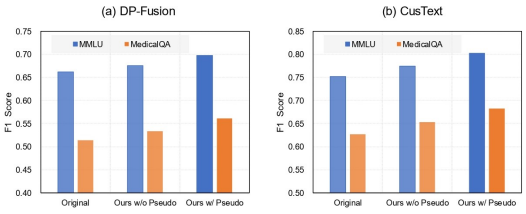


Figure 5: **Ablation study on pseudo prompt generation.** We conduct the ablation study for DP-Fusion and CusText. The genuine and pseudo requests are set as $n_r = 5000$ and $n_p = 5000$, respectively.

8 Conclusion

We presented a privacy amplification framework for black-box LLM inference that strengthens privacy beyond local prompt perturbation. By anonymously shuffling locally privatized prompts and augmenting batches with pseudo prompts, our framework enlarges the anonymity set and enables weaker local perturbation under the same end-to-end privacy guarantee. We further developed a mechanism-aware accountant that preserves likelihood-ratio moments of the underlying privatizer, providing tractable guarantees for both autoregressive and token-replacement mechanisms without enumerating the exponentially large output space. Empirical analysis demonstrates that our framework: (1) improves F1 by 8.2%–10.2% on average over existing DP mechanisms under the same privacy budget; (2) achieves over $2\times$ stronger privacy amplification than generic accounting when the local $\epsilon \geq 15$; and (3) improves F1 by 12.4% for DP-Fusion and 7.7% for CusText as the batch expands through pseudo-prompt augmentation.

AI USE STATEMENT

We used generative AI tools to assist in improving the clarity and readability of the manuscript and checking mathematical proofs. The authors take full responsibility for the final content of this work, including all claims, experiment results, and conclusions.

REFERENCES

Marah Abdin, Jyoti Aneja, Hany Awadalla, et al. Phi-3 technical report: A highly capable language model locally on your phone, 2024. URL https://arxiv.org/abs/2404.14219.

Ahmed Musa Awon, Yun Lu, Shera Potka, and Alex Thomo. Clusant: Differentially private and semantically coherent text sanitization. In Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pp. 3676–3693, 2025.

Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In Annual International Cryptology Conference, pp. 638–667. Springer, 2019.

Richard Barnes, Karthikeyan Bhargavan, Benjamin Lipp, and Christopher A Wood. Hybrid public key encryption. RFC 9180, 2022.

Junying Chen, Zhenyang Cai, Ke Ji, Xidong Wang, Wanlong Liu, Rongsheng Wang, Jianye Hou, and Benyou Wang. Huatuogpt-o1, towards medical complex reasoning with llms, 2024. URL https://arxiv.org/abs/2412.18925.

Sai Chen, Fengran Mo, Yanhao Wang, Cen Chen, Jian-Yun Nie, Chengyu Wang, and Jamie Cui. A customized text sanitization mechanism with differential privacy. In Findings of the Association for Computational Linguistics: ACL 2023, pp. 5747–5758, 2023.

Chun Jie Chong, Chenxi Hou, Zhihao Yao, and Seyed Mohammadjavad Seyed Talebi. Casper: Prompt sanitization for protecting user privacy in web-based large language models. In 2025 IEEE 12th International Conference on Cyber Security and Cloud Computing (CSCloud), pp. 122–133. IEEE, 2025.

Marcin Chrapek, Marcin Copik, Etienne Mettaz, and Torsten Hoefler. Confidential llm inference: Performance and cost across cpu and gpu tees. In 2025 IEEE International Symposium on Workload Characterization (IISWC), pp. 84–98. IEEE, 2025.

Minxin Du, Xiang Yue, Sherman SM Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. arXiv preprint arXiv:2309.06746, 2023.

Cynthia Dwork. Differential privacy. In International colloquium on automata, languages, and programming, pp. 1–12. Springer, 2006.

Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 2468–2479. SIAM, 2019.

Vitaly Feldman, Audra McMillan, and Kunal Talwar. Stronger privacy amplification by shuffling for rényi and approximate differential privacy. In Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 4966–4981. SIAM, 2023.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. In Proceedings of the 13th international conference on web search and data mining, pp. 178–186, 2020.

Antonious M Girgis, Deepesh Data, Suhas Diggavi, Ananda Theertha Suresh, and Peter Kairouz. On the renyi differential privacy of the shuffle model. In Proceedings of the 2021 ACM SIGSAC conference on computer and communications security, pp. 2321–2341, 2021.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. Proceedings of the International Conference on Learning Representations (ICLR), 2021.

Antti Koskela, Mikko A Heikkilä, and Antti Honkela. Numerical accounting in the shuffle model of differential privacy. Transactions on machine learning research, 2023.

Peihua Mai, Ran Yan, Zhe Huang, Youjia Yang, and Yan Pang. Split-and-denoise: protect large language model inference with local differential privacy. In Proceedings of the 41st International Conference on Machine Learning, ICML'24. JMLR.org, 2024.

Jimit Majmudar, Christophe Dupuy, Charith Peris, Sami Smaili, Rahul Gupta, and Richard Zemel. Differentially private decoding in large language models. arXiv preprint arXiv:2205.13621, 2022.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. The limits of word level differential privacy. In Findings of the Association for Computational Linguistics: NAACL 2022, pp. 867–881, 2022.

Ilya Mironov. Rényi differential privacy. In 2017 IEEE 30th computer security foundations symposium (CSF), pp. 263–275. IEEE, 2017.

Qi Pang, Jinhao Zhu, Helen Möllering, Wenting Zheng, and Thomas Schneider. Bolt: Privacy-preserving, accurate and efficient inference for transformers. In 2024 IEEE Symposium on Security and Privacy (SP), pp. 4753–4771. IEEE, 2024.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Supraja Sridhara, Andrin Bertschi, Benedict Schlüter, Mark Kuhne, Fabio Aliberti, and Shweta Shinde. {ACAI}: Protecting accelerator execution with arm confidential computing architecture. In 33rd USENIX Security Symposium (USENIX Security 24), pp. 3423–3440, 2024.

Robin Staab, Mark Vero, Mislav Balunovic, and Martin Vechev. Language models are advanced anonymizers. In International Conference on Learning Representations, volume 2025, pp. 98558–98598, 2025.

Pengcheng Su, Haibo Cheng, and Ping Wang. Decomposition-based optimal bounds for privacy amplification via shuffling. In 2026 IEEE Symposium on Security and Privacy (SP), pp. 2191–2210. IEEE, 2026.

Shun Takagi and Seng Pei Liew. Analysis of shuffling beyond pure local differential privacy. Proceedings of the ACM on Management of Data, 4(2 (PODS):1–26, 2026.

Rushil Thareja, Preslav Nakov, Praneeth Vepakomma, and Nils Lukas. Dp-fusion: Token-level differentially private inference for large language models. In International Conference on Learning Representations, volume 2026, pp. 151088–151120, 2026.

Rahul Krishna Thomas, Louai Zahran, Erica Choi, Akilesh Potti, Micah Goldblum, and Arka Pal. Hidden no more: Attacking and defending private third-party llm inference. In ICML, volume 267, pp. 59434–59469, 2025.

Martin Thomson and Christopher A Wood. Rfc 9458: Oblivious http, 2024.

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

Shaowei Wang, Yun Peng, Jin Li, Zikai Wen, Zhipeng Li, Shiyu Yu, Di Wang, and Wei Yang. Privacy amplification via shuffling: Unified, simplified, and tightened. Proc. VLDB Endow., 17 (8):1870–1883, April 2024. ISSN 2150-8097.

Tianshi Xu, Wen-jie Lu, Jiangrui Yu, Yi Chen, Chenqi Lin, Runsheng Wang, and Meng Li. Breaking the layer barrier: Remodeling private transformer inference with hybrid {CKKS} and {MPC}. In 34th USENIX Security Symposium (USENIX Security 25), pp. 2653–2672, 2025.

Tianyu Yang, Xiaodan Zhu, and Iryna Gurevych. Robust utility-preserving text anonymization based on large language models. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 28922–28941, 2025.

Andrew C Yao. Protocols for secure computations. In 23rd annual symposium on foundations of computer science (sfcs 1982), pp. 160–164. IEEE, 1982.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for {Transformer-Based} generative models. In 16th USENIX symposium on operating systems design and implementation (OSDI 22), pp. 521–538, 2022.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman SM Chow. Differential privacy for text analytics via natural text sanitization. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pp. 3853–3866, 2021.

Juhua Zhang, Zhiliang Tian, Minghang Zhu, Yiping Song, Taishu Sheng, Siyi Yang, Qiunan Du, Xinwang Liu, Minlie Huang, and Dongsheng Li. Dyntext: semantic-aware dynamic text sanitization for privacy-preserving llm inference. In Findings of the Association for Computational Linguistics: ACL 2025, pp. 20243–20255, 2025.

Lele Zheng, Chao Zhang, Feiyang Yuan, Ke Cheng, Tao Zhang, Anxiao Song, and Yulong Shen. Dp3: Differentially private prompt perturbation for multi-turn llm inference. In Findings of the Association for Computational Linguistics: ACL 2026, pp. 18538–18552, 2026.

Yuqing Zhu, Jinshuo Dong, and Yu-Xiang Wang. Optimal accounting of differential privacy via characteristic function. In International Conference on Artificial Intelligence and Statistics, pp. 4782–4817. PMLR, 2022.

A FRAMEWORK

A.1 ALGORITHM

Algorithm 1 summarizes the pseudo-prompt augmentation procedure.

Algorithm 1 PSEUDOPROMPTSHUFFLE

Input: private prompts $X = (x_1, \ldots, x_{n_r})$; number of pseudo prompts n_p; public pseudo-prompt distribution Π; local privatizer $\mathcal{M}$.
Output: shuffled privatized prompts.

for $j = 1, \ldots, n_p$ **do**
  Sample $\tilde{x}_j \sim \Pi$ independently of X
end for
$\widetilde{X} \leftarrow (x_1, \ldots, x_{n_r}, \tilde{x}_1, \ldots, \tilde{x}_{n_p})$
for $i = 1, \ldots, n_r + n_p$ **do**
  $y_i \leftarrow \mathcal{M}(\widetilde{x}_i)$ using independent randomness
end for
return $\mathsf{Shuf}(y_1, \ldots, y_{n_r+n_p})$

A.2 PSEUDO-PROMPT AUGMENTATION

We consider pseudo-prompt augmentation under two settings, depending on whether the local privatizer exposes public information from the original prompt.

A.2.1 PROMPTS WITHOUT PUBLIC INFORMATION

When the entire prompt is treated as private, pseudo prompts are sampled independently of the genuine batch from a fixed public distribution Π:

$$\widetilde{x}_j \overset{\text{i.i.d.}}{\sim} \Pi, \quad j \in [n_p].$$

Each pseudo prompt is processed by an independent execution of the same local privatizer, $\widetilde{y}_j \sim \mathcal{M}(\widetilde{x}_j)$, and the resulting prompts are mixed with the genuine privatized prompts before shuffling.

A.2.2 PROMPTS WITH PUBLIC INFORMATION

Public-context adjacency and grouping. For mechanisms that leave part of the prompt publicly observable, we denote each prompt as $x = (c, s)$, where c denotes public context and s denotes the protected content. Neighboring inputs are defined as $(c, s) \sim (c, s')$, with public context c fixed. The shuffler receives the public context together with the privatized report and partitions the batch into context-compatible groups.

Pseudo-prompt allocation. Let $n_r(c)$ denote the number of genuine requests associated with context c. We allocate a fixed pseudo-prompt budget n_p across the public contexts proportionally:

$$n_p(c) = n_p \frac{n_r(c)}{n_r}, \quad \sum_c n_p(c) = n_p, \tag{3}$$

with rounding chosen to preserve the total budget.

Context-conditioned pseudo-prompt generation. For each context c, the shuffler samples protected content from a public distribution:

$$\widetilde{s}_{c,j} \sim \Pi_c, \quad \widetilde{x}_{c,j} = (c, \widetilde{s}_{c,j}), \quad j \in [n_p(c)],$$

where Π_c depends only on public information and is independent of the genuine protected values. Each pseudo prompt is processed using the same privatization mechanism as genuine prompts:

$$\widetilde{y}_{c,j} \sim \mathcal{M}(\widetilde{x}_{c,j}).$$

The resulting pseudo prompts are then mixed with genuine prompts in the same context group and shuffled before being forwarded to the service provider.

A.3 SECURE COMMUNICATION

To secure communication among users, shuffler, and LLM service provider, we design a lightweight cryptographic protocol inspired by Oblivious HTTP Thomson & Wood (2024). The protocol has two goals: (1) to protect users' sensitive content from the shuffle; and (2) to deliver a shuffled batch of requests to the provider without revealing their originating users.

Cryptographic setup. We use Hybrid Public Key Encryption (HPKE) Barnes et al. (2022), which combines public-key key establishment with authenticated symmetric encryption. The gateway publishes an authenticated public key pk_P, shared by all users within a key epoch. Request and response payloads are padded to public byte lengths within each context group.

Encrypted submission. Let c denote the disclosed public context, with $c = \perp$ when no public context is disclosed. Given public context c, user i locally privatizes its prompt as $y_i \sim M_c(s_i)$. When $c = \perp$, s_i denotes the entire prompt and $M_\perp(s_i) = M(s_i)$. The user then samples a fresh response key $k_i \leftarrow \{0,1\}^{128}$. For public batch identifier b and context c, the user sends (b, c, C_i) to the shuffler, with:

$$C_i = \mathsf{HPKE.Enc}_{pk_P}\big(\mathrm{pad}(y_i, k_i); \mathsf{aad} = (b, c)\big),$$

where HPKE.Enc denotes HPKE encryption with fresh randomness, pad serializes and pads the payload to the prescribed length, and aad denotes authenticated but unencrypted associated data.

Augmentation and shuffling. The shuffler samples pseudo inputs from Π_c, privatizes them using M_c, and encrypts them using the same format and fresh response keys. When $c = \perp$, we use a fixed public distribution $\Pi_\perp = \Pi$ independent of genuine prompts. It uniformly permutes the combined batch and forwards it through a shared gateway connection. The permutation map remains local to S.

Encrypted responses. After decrypting each request, the gateway computes $z_i = \mathrm{LLM}(y_i)$ and returns:

$$D_i = \mathsf{AEAD.Enc}_{k_i}\big(\nu_i, \mathrm{pad}(z_i); \mathsf{aad} = (b, c, H(C_i))\big),$$

where AEAD.Enc denotes authenticated encryption using AES-128-GCM, ν_i denotes the fresh nonce, and H is SHA-256.

All responses, including those for pseudo requests, are returned by shuffled position. The shuffler forwards genuine responses to their users and discards pseudo responses locally without revealing which are retained. Users decrypt responses using their respective keys.

B MECHANISM-AWARE SHUFFLING BOUND

B.1 ALGORITHMS

Algorithm 2 provides the implementation details for computing the shuffling bound, using Algorithm 3 as subroutine.

B.2 TECHNICAL DETAILS

B.2.1 INTEGRATED MOMENT-TAIL CONVERSION

We provide the detailed conversion from shuffled likelihood-ratio moments to an (ϵ, δ)-DP guarantee.

Let $\widetilde{P}_X$ and $\widetilde{P}_{X'}$ denote the output distributions of the shuffled mechanism $\mathcal{S}_n$ on neighboring datasets $X \sim X'$. Assuming $\widetilde{P}_X \ll \widetilde{P}_{X'}$, define the shuffled likelihood ratio as:

$$R^{\mathrm{shuf}}_{X,X'}(z) = \frac{\widetilde{P}_X(z)}{\widetilde{P}_{X'}(z)}.$$

Using the hockey-stick characterization of differential privacy, we have:

$$H_{e^\epsilon}(\widetilde{P}_X \| \widetilde{P}_{X'}) = \int_{e^\epsilon}^{R^{\mathrm{shuf}}_{\max}} \Pr_{\widetilde{P}_{X'}}\left[R^{\mathrm{shuf}}_{X,X'} \geq s\right] ds, \tag{4}$$

where $R^{\mathrm{shuf}}_{\max}$ is an upper bound on the shuffled likelihood ratio.

Algorithm 2 MECHANISMAWAREPRIVACYACCOUNTANT

Input: number of users n; output length T; maximum moment order $\alpha_{\max}$; overlap coefficient γ; target δ; privatizer type $\mathsf{type} \in \{\mathsf{rep}, \mathsf{gen}\}$; mechanism parameters $\{K_t\}_{t=1}^T$ or $\{B_t\}_{t=1}^T$.
Output: privacy parameter $\epsilon_{\text{int}}(\delta)$.

$\overline{M}_0 \leftarrow 1, \overline{M}_1 \leftarrow 1$
for $k = 2, \dots, \alpha_{\max}$ **do**
  for $t = 1, \dots, T$ **do**
    if $\mathsf{type} = \mathsf{rep}$ **then**
      $A_{k,t} \leftarrow \sup_{x \sim x'} \sum_y K_t(y \mid x_t)^k K_t(y \mid x_t')^{1-k}$
    else
      $A_{k,t} \leftarrow \psi_k(B_t)$ ▷ equation 6
    end if
  end for
  $\overline{M}_k \leftarrow \prod_{t=1}^T A_{k,t}$
end for
for $\alpha = 2, \dots, \alpha_{\max}$ **do**
  $\overline{L}_\alpha^{\text{shuf}} \leftarrow$ SHUFFLEDMOMENT$(n, \gamma, \alpha, \{\overline{M}_k\}_{k=0}^\alpha)$
end for
$x_{\max} \leftarrow \log(1/\gamma)$
$\mathcal{A} \leftarrow \{2, \dots, \alpha_{\max}\}$
Define $g(x) \leftarrow \min\left\{0, \min_{\alpha \in \mathcal{A}} \left[\overline{L}_\alpha^{\text{shuf}} + (1-\alpha)x\right]\right\}$
Define $\Delta(\epsilon) \leftarrow \int_\epsilon^{x_{\max}} e^{g(x)}\, dx$
$\epsilon_{\text{int}}(\delta) \leftarrow \inf\{\epsilon \in [0, x_{\max}] : \Delta(\epsilon) \le \delta\}$ ▷ computed by bisection
return $\epsilon_{\text{int}}(\delta)$

Algorithm 3 SHUFFLEDMOMENT

Input: n, γ, order α, local moments $\{\overline{M}_k\}_{k=0}^\alpha$.
Output: shuffled log-moment bound $\overline{L}_\alpha^{\text{shuf}}$.

$F_\alpha(z) \leftarrow \sum_{k=0}^\alpha \frac{\overline{M}_k}{k!} z^k$
$V_\alpha \leftarrow 0$
for $b = 0, \dots, n-1$ **do**
  $V_\alpha \leftarrow V_\alpha + \binom{n-1}{b} \gamma^b (1-\gamma)^{n-1-b} \frac{\alpha!}{(b+1)^\alpha} [z^\alpha] F_\alpha(z)^{b+1}$
end for
$\overline{L}_\alpha^{\text{shuf}} \leftarrow \min\left\{\log \overline{M}_\alpha, \log V_\alpha\right\}$
return $\overline{L}_\alpha^{\text{shuf}}$

For an integer order $\alpha \ge 2$, define the worst-case shuffled log moment as:

$$L_\alpha^{\text{shuf}} = \sup_{X \sim X'} \log \mathbb{E}_{Z \sim \widetilde{P}_{X'}} \left[\left(R_{X,X'}^{\text{shuf}}(Z)\right)^\alpha\right].$$

Suppose that the mechanism-aware accountant provides valid upper bounds:

$$L_\alpha^{\text{shuf}} \le \overline{L}_\alpha^{\text{shuf}}, \qquad \alpha \in \mathcal{A},$$

where $\mathcal{A} \subseteq \{2, 3, \dots\}$ is the set of moment orders used for accounting.

By Markov's inequality, it holds that:

$$\Pr_{\widetilde{P}_{X'}}\left[R_{X,X'}^{\text{shuf}} \ge s\right] \le e^{\overline{L}_\alpha^{\text{shuf}}} s^{-\alpha}.$$

Moreover, since $\mathbb{E}_{\widetilde{P}_{X'}}[R^{\mathrm{shuf}}_{X,X'}] = 1$, we also have:

$$\Pr_{\widetilde{P}_{X'}}\left[R^{\mathrm{shuf}}_{X,X'} \geq s\right] \leq \frac{1}{s}.$$

Combining these bounds gives:

$$H_{e^\epsilon}(\widetilde{P}_X \| \widetilde{P}_{X'}) \leq \int_{e^\epsilon}^{R^{\mathrm{shuf}}_{\max}} \min\left\{\frac{1}{s}, \min_{\alpha\in\mathcal{A}} e^{\overline{L}^{\mathrm{shuf}}_\alpha} s^{-\alpha}\right\} ds \triangleq \delta_{\mathrm{int}}(\epsilon).$$

The resulting privacy guarantee is:

$$\epsilon_{\mathrm{int}}(\delta) = \inf\left\{\epsilon : \delta_{\mathrm{int}}(\epsilon) \leq \delta\right\}.$$

In implementation, the integral is evaluated in the log-likelihood-ratio domain. Let $x = \log s$. Then the integrand is determined by the lower envelope of the affine functions:

$$0, \qquad \overline{L}^{\mathrm{shuf}}_\alpha + (1-\alpha)x, \quad \alpha \in \mathcal{A},$$

which permits efficient piecewise integration followed by a one-dimensional search over ϵ.

B.2.2 Derivation of the Mechanism-Aware Shuffled Moment Bound

We next derive the shuffled moment bound used in Section 5.3. For neighboring prompts $x \sim x'$, recall the order-k local likelihood-ratio moment:

$$M_k(x, x') = \mathbb{E}_{Y\sim P_{x'}}\left[\left(\frac{P_x(Y)}{P_{x'}(Y)}\right)^k\right], \qquad k \geq 2,$$

and let $\overline{M}_k$ be a uniform upper bound over neighboring inputs, with $\overline{M}_0 = \overline{M}_1 = 1$.

Overlap coefficient. To capture the common component shared by local output distributions, we define:

$$\gamma = \inf_{u,v\in\mathcal{X}} \operatorname*{ess\,inf}_{y:P_v(y)>0} \frac{P_u(y)}{P_v(y)}. \tag{5}$$

We assume $\gamma > 0$. Equivalently, for any $u, v \in \mathcal{X}$, there exists a distribution $G_{u,v}$ such that:

$$P_u = \gamma P_v + (1-\gamma)G_{u,v}.$$

If $\mathcal{M}$ satisfies ϵ_0-LDP over the admissible prompt domain, then $\gamma \geq e^{-\epsilon_0}$.

Consider neighboring datasets:

$$X = (x, x_2, \ldots, x_n), \qquad X' = (x', x_2, \ldots, x_n),$$

and let $Q = P_{x'}$. For each unchanged report $j \geq 2$, the overlap condition implies a decomposition:

$$P_{x_j} = \gamma Q + (1-\gamma)G_j.$$

Introduce independent indicators specifying whether each unchanged report is drawn from the common component Q. Their total number satisfies:

$$B \sim \mathrm{Binomial}(n-1, \gamma).$$

Conditioned on $B = b$, let $m = b + 1$. After revealing the residual components, the privacy-relevant experiment reduces to hiding one report drawn from P_x among $m-1$ reports drawn from Q. Under the reference dataset, all m reports are drawn from Q. The conditional shuffled likelihood ratio is therefore:

$$S_m = \frac{1}{m}\sum_{i=1}^m r(Y_i), \quad Y_i \overset{\text{i.i.d.}}{\sim} Q, \quad r(y) = \frac{P_x(y)}{Q(y)}$$

For integer $\alpha \geq 2$, multinomial expansion gives:

$$\mathbb{E}[S_m^\alpha] = \frac{\alpha!}{m^\alpha}[z^\alpha]\left(\sum_{k=0}^\alpha \frac{M_k(x,x')}{k!} z^k\right)^m.$$

Replacing the pair-specific moments by their uniform upper bounds and averaging over B yields:

$$\overline{L}_\alpha^{\text{shuf}} = \min\left\{\log \overline{M}_\alpha,\ \log\left[\sum_{b=0}^{n-1}\binom{n-1}{b}\gamma^b(1-\gamma)^{n-1-b}\frac{\alpha!}{(b+1)^\alpha}[z^\alpha]\left(\sum_{k=0}^{\alpha}\frac{\overline{M}_k}{k!}z^k\right)^{b+1}\right]\right\}.$$

The first term follows from data processing, since shuffling cannot increase the divergence beyond that of the local report, while the second term captures the amplification induced by anonymity.

B.2.3 Mechanism-Specific Local Moment Bounds

We now detail how the local moment bounds $\{\overline{M}_k\}$ are obtained from the sequential structure of prompt privatization.

Suppose the local privatizer admits the factorization:

$$P_x(y_{1:T}) = \prod_{t=1}^{T} p_{x,t}(y_t \mid y_{<t}).$$

For each step t, define the conditional moment envelope:

$$A_{k,t} = \sup_{x\sim x'}\sup_{h}\sum_{y} p_{x,t}(y \mid h)^k p_{x',t}(y \mid h)^{1-k},$$

where h denotes a realization of the previous output $Y_{<t}$. Repeated conditioning yields:

$$M_k(x,x') \le \prod_{t=1}^{T} A_{k,t},$$

and we therefore use:

$$\overline{M}_k = \prod_{t=1}^{T} A_{k,t}.$$

Token replacement. For token-wise replacement, $p_{x,t}(y \mid h) = K_t(y \mid x_t)$, does not depend on the previous output history. Hence, we have:

$$A_{k,t}^{\text{rep}} = \sup_{x\sim x'}\sum_{y} K_t(y \mid x_t)^k K_t(y \mid x_t')^{1-k}.$$

Thus, the local moment can be evaluated directly from the replacement channel without enumerating sequence-level outputs.

For independent replacement across protected positions, it holds that:

$$\overline{M}_k = \prod_{t\in I} A_{k,t}^{\text{rep}},$$

where I denotes the set of privatized positions. This form is used for token-replacement mechanisms such as CusText.

Autoregressive generation. For autoregressive mechanisms, the conditional distribution $p_{x,t}(y \mid h)$ depends on the previously generated sequence, making direct optimization over all histories generally intractable. We instead characterize the range of the conditional privacy loss as:

$$B_t = \sup_{x\sim x',h}\left\{\sup_y \ell_t(y;h,x,x') - \inf_y \ell_t(y;h,x,x')\right\}, \quad \ell_t(y;h,x,x') = \log\frac{p_{x,t}(y \mid h)}{p_{x',t}(y \mid h)}$$

A bounded-range privacy-loss argument gives:

$$A_{k,t}^{\text{gen}} \le \psi_k(B_t),$$

where $\psi_k(0) = 1$ and, for $B > 0$,

$$\psi_k(B) = \max_{u\in[0,B]}\left[\frac{e^u-1}{e^B-1}e^{k(B-u)} + \frac{e^B-e^u}{e^B-1}e^{-ku}\right]. \tag{6}$$

Consequently, it holds that:

$$\overline{M}_k \le \prod_{t=1}^{T} \psi_k(B_t).$$

The function $\psi_k(B)$ is tight given only the width B of the privacy-loss range: the extremal distribution is supported on the two endpoints of the interval. This allows autoregressive mechanisms to be summarized through per-step range bounds rather than through explicit enumeration of all possible generation histories.

B.3 Proof of Theorem 6.1

Proof. Fix neighboring datasets

$$X = (x, x_2, \dots, x_n), \qquad X' = (x', x_2, \dots, x_n).$$

Local moments. By the sequential factorization,

$$\frac{P_x(Y_{1:T})}{P_{x'}(Y_{1:T})} = \prod_{t=1}^{T} \frac{p_{x,t}(Y_t \mid Y_{<t})}{p_{x',t}(Y_t \mid Y_{<t})}.$$

For every history h,

$$\mathbb{E}_{p_{x',t}(\cdot|h)} \left[\left(\frac{p_{x,t}(Y_t \mid h)}{p_{x',t}(Y_t \mid h)} \right)^k \right] \le A_{k,t}.$$

Repeated conditioning therefore gives

$$M_k(x, x') \le \prod_{t=1}^{T} A_{k,t} = \overline{M}_k.$$

The constructions in Section 5.4 provide valid $A_{k,t}$ for both token replacement and autoregressive generation.

Shuffled moments. Let $Q = P_{x'}$. By the definition of γ, for every unchanged user $j \ge 2$ there exists a distribution G_j such that

$$P_{x_j} = \gamma Q + (1 - \gamma) G_j.$$

Introduce the corresponding common-component indicators. Their number B follows $\mathrm{Binomial}(n - 1, \gamma)$. Revealing these indicators and the residual components can only increase Rényi divergence.

Conditioned on $B = b$, let $m = b + 1$. The privacy-relevant shuffled experiment consists of one P_x report hidden uniformly among $m - 1$ reports from Q. Under the reference dataset, all m reports follow Q. Hence, with

$$r(y) = \frac{P_x(y)}{Q(y)},$$

the conditional likelihood ratio is

$$S_m = \frac{1}{m} \sum_{i=1}^{m} r(Y_i), \qquad Y_i \overset{\text{i.i.d.}}{\sim} Q.$$

For integer $\alpha \ge 2$, multinomial expansion yields

$$\mathbb{E}[S_m^\alpha] = \frac{\alpha!}{m^\alpha} [z^\alpha] \left(\sum_{k=0}^{\alpha} \frac{M_k(x, x')}{k!} z^k \right)^m.$$

Using $M_k(x, x') \le \overline{M}_k$ and averaging over $B \sim \mathrm{Binomial}(n - 1, \gamma)$ gives the shuffling term computed in Algorithm 2. Moreover, since shuffling is post-processing of the unshuffled local reports,

$$L_\alpha^{\text{shuf}} \le \log \overline{M}_\alpha.$$

Taking the minimum of the two bounds gives

$$L_\alpha^{\text{shuf}} \le \overline{L}_\alpha^{\text{shuf}}$$

for every $\alpha = 2, \ldots, \alpha_{\max}$.

DP conversion. For the shuffled likelihood ratio $R = R^{\text{shuf}}_{X,X'}$, we have

$$\mathbb{E}_{\widetilde{P}_{X'}}[R] = 1, \qquad \mathbb{E}_{\widetilde{P}_{X'}}[R^\alpha] \le e^{\overline{L}^{\text{shuf}}_\alpha}.$$

Thus, Markov's inequality gives, for every $s \ge 1$,

$$\Pr[R \ge s] \le \min\left\{\frac{1}{s}, \min_\alpha e^{\overline{L}^{\text{shuf}}_\alpha} s^{-\alpha}\right\}.$$

Substituting this bound into the hockey-stick tail representation in equation 4 gives

$$H_{e^\epsilon}(\widetilde{P}_X \| \widetilde{P}_{X'}) \le \delta_{\text{int}}(\epsilon).$$

Therefore, for $\epsilon = \epsilon_{\text{int}}(\delta)$ returned by Algorithm 2,

$$H_{e^\epsilon}(\widetilde{P}_X \| \widetilde{P}_{X'}) \le \delta$$

for every neighboring $X \sim X'$. By the hockey-stick characterization of differential privacy, $\mathcal{S}_n$ is $(\epsilon_{\text{int}}(\delta), \delta)$-DP. □

B.4 Privacy Accounting with Public Context

Theorem 6.1 and Corollary 6.2 extend to public context by applying the accountant within each context group. Denote the prompt as $x = (c, s)$, where c is public and s is protected. Adjacency changes one genuine prompt's protected content while fixing all public contexts. For each fixed public context c, define conditional privatizer $M_c(s) = M(c, s)$. Theorem 6.1's assumptions must hold for M_c over all admissible protected inputs s.

Corollary B.1 provides the privacy guarantee for jointly releasing shuffled prompts across context groups.

Corollary B.1 (Privacy with public context)**.** *Fix the public-context vector C, and define adjacency by changing the protected content of one genuine prompt while preserving C. For each context $c \in \mathcal{C}$, suppose M_c satisfies the assumptions of Theorem 6.1, with overlap γ_c and local moment bounds $\overline{M}_{k,c}$ on a domain containing all genuine and pseudo inputs.*

Conditional on C, let pseudo-prompt counts depend only on public information and pseudo inputs be sampled independently from public distributions, independently of genuine protected content. Process and shuffle prompts within each group using independent randomness across prompts and groups. If $\epsilon_c(\delta)$ is returned by Algorithm 2 using $n(c) = n_r(c) + n_p(c)$, γ_c, and $\{\overline{M}_{k,c}\}$, then the joint release satisfies $(\max_{c\in\mathcal{C}} \epsilon_c(\delta),\ \delta)$ -DP.

Proof. Fix adjacent datasets X and X' with public context C. Their differing prompt belongs to a single group, denoted by $c^\star$. Conditional on C, the pseudo-prompt counts and sampling distributions are identical under both datasets. Applying Theorem 6.1 and Corollary 6.2 within each group gives $(\epsilon_c(\delta), \delta)$-DP with respect to changes in one genuine protected input.

Let Q and Q' denote the output distributions of group $c^\star$ under X and X'. All other groups have the same joint output distribution R under both datasets. Independence across groups therefore gives full output distributions $Q \otimes R$ and $Q' \otimes R$, apart from the identical public contexts. Set $\epsilon = \max_{c\in\mathcal{C}} \epsilon_c(\delta)$. For any measurable event A, write $A_r = \{z : (z, r) \in A\}$. Then

$$(Q \otimes R)(A) = \int Q(A_r)\, R(dr) \le \int \left[e^{\epsilon_{c^\star}(\delta)} Q'(A_r) + \delta\right] R(dr) \le e^\epsilon (Q' \otimes R)(A) + \delta.$$

The same argument applies with X and X' interchanged. Releasing the fixed public contexts incurs no additional privacy loss under this adjacency, completing the proof. □

Adjacency invariant grouping. Corollary B.1 requires the grouping context to be independent of protected content. We show that our construction satisfies this requirement.

Let each prompt be written as $x = (c, s)$, where c denotes the public context and s denotes the protected content. Let $G : \mathcal{X} \to \mathcal{G}$ denote the grouping rule used by the shuffler. We consider

adjacency over protected content $(c, s) \sim (c, s')$, where c is fixed and s may vary arbitrarily within its admissible domain. A grouping rule is adjacency invariant if

$$x \sim x' \implies G(x) = G(x').$$

In our framework, G depends only on public information: $G(c, s) = g(c)$ for some deterministic g. Therefore, changing the protected content does not affect the group assignment.

B.5 Trade-off between Privacy and Computation

Let $n = n_r + n_p$, where $n_r \geq 1$ is fixed and n_p is chosen independently of the protected inputs. For requests with public information, the analysis applies separately to each group of public context.

Proposition B.2 (Privacy under batch expansion)**.** *Under the assumptions of Theorem 6.1 and Corollary 6.2, let $\gamma \in (0, 1]$ be a valid uniform overlap coefficient on the admissible input domain, and let $v = \overline{M}_2 - 1$. Then it holds that:*

$$\sup_{X \sim X'} D_2(S_n(X) \,\|\, S_n(X')) \leq \rho_2(n) = \log\left(1 + v\frac{1 - (1-\gamma)^n}{n\gamma}\right),$$

where D_2 denotes order-2 Rényi divergence. The bound $\rho_2(n)$ is nonincreasing in n.

Proof. Fix an ordered neighboring pair differing in one genuine input, x versus x', and let $Q = P_{x'}$ and $r = P_x/Q$. By the overlap assumption, every unchanged report distribution, including the pseudo-report mixture $\int P_u \, \Pi(du)$, contains Q with mixture weight γ.

The common-component decomposition from Theorem 6.1 gives $B \sim \mathrm{Binomial}(n-1, \gamma)$ unchanged reports drawn from Q. Conditional on $B = b$, the likelihood ratio of the augmented experiment is:

$$R_m = \frac{1}{m}\sum_{i=1}^{m} r(Y_i), \qquad m = b + 1, \qquad Y_i \overset{\text{i.i.d.}}{\sim} Q.$$

Since $\mathbb{E}_Q[r] = 1$ and $\mathrm{Var}_Q(r) \leq \overline{M}_2 - 1 = v$, we have:

$$\mathbb{E}[R_m^2] = 1 + \frac{\mathrm{Var}_Q(r)}{m} \leq 1 + \frac{v}{m}.$$

Averaging over B and applying data processing yields:

$$\exp(D_2(S_n(X) \,\|\, S_n(X'))) \leq 1 + v\,\mathbb{E}\left[\frac{1}{B+1}\right].$$

Furthermore, it follows that:

$$\mathbb{E}\left[\frac{1}{B+1}\right] = \int_0^1 (1 - \gamma + \gamma t)^{n-1}\, dt = \frac{1 - (1-\gamma)^n}{n\gamma}.$$

Taking logarithms and the supremum over neighboring pairs proves the bound. For fixed γ and $v \geq 0$, the integrand is non-increasing in n, so $\rho_2(n)$ is also non-increasing. □

Computational cost. Let $\kappa = n/n_r$ denote the batch expansion factor. The provider processes κn_r requests. Under an additive per-request cost model, the expected request-processing work is:

$$\mathcal{C}(n_p) = \mathcal{C}(0) + n_p c_p,$$

where $\mathcal{C}(0)$ is the cost of processing the n_r genuine requests, and c_p accounts for construction, privatization, and LLM response generation cost of each pseudo-prompt. Shuffling additionally requires $O(n)$ work. For fixed privatizer settings and maximum moment order A, the local moment profile can be reused, while computing the shuffled moment profile requires $O(nA^2)$ work. Thus, the total cost grows linearly with batch size.

Proposition B.2 gives $\rho_2(n) \leq v/(\gamma\kappa n_r)$, while total cost grows linearly with κ. This characterizes a tunable trade-off between complexity and privacy, where pseudo prompts strengthen privacy amplification through additional computation without waiting for more genuine requests.

C EXPERIMENT

C.1 EXPERIMENT SETTING

The experiments are conducted on a 96-core Ubuntu Linux 20.04 server with 128GB RAM and 6 L40 driver. Each reported experiment result is an average of 4 independent runs. The maximum sequence length is $T = 100$ tokens.

C.1.1 DP MECHANISMS

For DP-Prompt and DP-Fusion, we use Phi-3.5 (`microsoft/Phi-3.5-mini-instruct`) Abdin et al. (2024) as the underlying paraphrasing model. For CusText, we use input token embeddings of GPT-2 (`openai-community/gpt2`) Radford et al. (2019) to compute the similarities used in constructing replacement sets for substitution. The replacement set size for CusText is set as 20.

DP-Prompt privatizes the entire prompt, whereas DP-Fusion and CusText privatize sensitive content while preserving the public template. For DP-Prompt, privacy budget ϵ corresponds to full-prompt adjacency. For DP-Fusion and CusText, the privacy budget ϵ is defined with respect to prompt-level replacement adjacency on the entire protected content, conditioned on a fixed public template.

To construct benchmark with protected content, we use GPT-5-mini offline to annotate candidate protected spans using the instruction below. Note that this model is used solely to construct the experimental annotations and is not part of the deployed privacy-preserving inference pipeline.

> Extract words or phrases from the question that indicate sensitive attributes, including: personal identifiers; contact details, locations, residency, citizenship, URLs, and digital footprints; age, race, ethnicity, religion, beliefs, sexuality, gender, third-person pronouns, dates, and festivals; employment, income, education, and company information; physical or mental health, disability, clinical findings, behavior, medications, drug classes, and biometric or genetic data; financial records, accounts, loans, taxes, contracts, trading algorithms, and confidential business information; legal matters, criminal records, ownership, consumer rights, law-enforcement jurisdiction and procedures; voting, union membership, surveillance, military service, and security clearance.
>
> Preserve the exact wording and all repeated occurrences. Do not paraphrase, alter abbreviations, or add information.
>
> Return only a JSON list: `["attribute 1", "attribute 2", ...]`.
>
> If no sensitive content is found, return `[]`.
>
> Question: {`question`}
>
> Sensitive attributes:

C.1.2 COMPUTATION OF OVERLAP COEFFICIENT

The overlap coefficient γ is computed for each DP mechanism as follows.

DP-Prompt. With logits clipped to $[\ell, u]$ before softmax at temperature $\tau > 0$, the probability ratio between any two inputs is at least $\exp(-2(u-\ell)/\tau)$ at every generation step and history. For generation token limit $T_{\max}$, we therefore use:

$$\gamma = \exp\left(-\frac{2T(u-\ell)}{\tau}\right).$$

DP-Fusion. For each component mixture $p = (1-\lambda)q + \lambda r$, where q and r are the public and private next-token distributions, we additionally enforce $e^{-\kappa}q(y) \le p(y) \le e^{\kappa}q(y)$ for a fixed public parameter $\kappa > 0$. We cap the original RDP-admissible weight λ_{RDP} by:

$$\lambda = \min\left\{\lambda_{\mathrm{RDP}}, \frac{1-e^{-\kappa}}{1-m}, \frac{e^{\kappa}-1}{M-1}\right\}, \quad m = \min_y \frac{r(y)}{q(y)}, \quad M = \max_y \frac{r(y)}{q(y)},$$

which holds after averaging component mixtures. Consequently, the overlap coefficient is lower bounded by:

$$\gamma = \exp(-2\kappa T).$$

CusText. For each group $\mathcal{A}_j$ of input tokens sharing replacement set $\mathcal{R}_j$, let $K_j(y \mid u)$ denote the replacement probability. During offline stage, we precompute:

$$\gamma_j = \min_{y \in \mathcal{R}_j} \frac{\min_{u \in \mathcal{A}_j} K_j(y \mid u)}{\max_{v \in \mathcal{A}_j} K_j(y \mid v)}.$$

During online accounting for public-context group c, independent replacement at protected positions I_c gives:

$$\gamma = \prod_{t \in I_c} \gamma_{j_t},$$

where j_t indexes the replacement set at position t, which is fixed within group c.

C.1.3 Pseudo-prompt Augmentation

DP-Prompt. We sample pseudo prompts uniformly with replacement from the question collection of the corresponding benchmark, namely MMLU and Medical-QA. The candidate collection $\mathcal{D}_{\text{pub}}$ is fixed before constructing genuine batches. Each sampled question is formatted using the same task instruction as genuine requests and privatized by DP-Prompt.

DP-Fusion. Each user sends the encrypted privatized prompt and plaintext public template to the shuffler. We use a fixed pretrained T5-base model Raffel et al. (2020) to fill protected slots sequentially, conditioning only on the public template and previously generated synthetic content. Each completed pseudo prompt is privatized using the same DP-Fusion baseline and parameters as genuine prompts in its group.

CusText. The shuffler receives encrypted sensitive content alongside the plaintext public template. we fix the token mapping used by CusText and group prompts by tokens retained without perturbation. T5-base Raffel et al. (2020) fills protected positions sequentially using only this public context and previously generated synthetic tokens. We then apply the original CusText replacement channel independently at each position.

Allocation and privacy accounting. Unless otherwise specified, each batch contains $n_r = 5000$ genuine prompts and $n_p = 5000$ pseudo prompts. For DP-Fusion and CusText, we allocate pseudo prompts proportionally to the genuine group size based on equation 3, using rounding determined solely by the public group counts.

C.1.4 Privacy Budget Calibration

For DP-Fusion and CusText, accounting is performed separately within each context group c using its actual augmented group size $n(c) = n_r(c) + n_p(c)$. The local privatization parameters are calibrated such that $\epsilon_c \leq \epsilon$ for every context group. Therefore, by Corollary B.1, the joint release satisfies $\max_c \epsilon_c \leq \epsilon$. In all experiments reporting a server-side privacy budget ϵ, this group-wise calibration is used for DP-Fusion and CusText.

C.1.5 Latency Analysis

We evaluate latency in terms of active processing time. We assume that the shuffler parallelizes API inference and pseudo-prompt generation across prompts. LLM inference latency is estimated using the average per-prompt query time measured without privacy protection.

C.1.6 Inference Attack

Prompt Distinguishing Attack. We consider an adversary that distinguishes which of two neighboring prompts, x or x', generated one unknown report in a shuffled batch. The remaining $n-1$ prompts are independently sampled from a fixed public background distribution Π_{bg}, independently of the target hypothesis. Let $Q(y) = \mathbb{E}_{u \sim \Pi_{\text{bg}}}[P_u(y)]$ denote the induced distribution of a privatized background report. The attacker knows the privatization mechanism, x, x', and Π_{bg}, but not the identities or realized prompts of the background reports. Under hypothesis H_0, one report follows P_x and the other $n-1$ reports follow Q; under H_1, the target report instead follows $P_{x'}$. After uniform

shuffling, the likelihood ratio for an observed batch $Z = \{y_1, \ldots, y_n\}$ is given by:

$$R(Z) = \frac{\sum_{i=1}^{n} P_x(y_i)/Q(y_i)}{\sum_{i=1}^{n} P_{x'}(y_i)/Q(y_i)},$$

which is used as the distinguishing score by the attacker.

Attribute Inference Attack. We evaluate whether the attacker can recover a sensitive attribute from the privatized prompt. For each candidate attribute $a \in \mathcal{A}$, the attacker computes the likelihood of the observed privatized report under the corresponding prompt $x(a)$:

$$P(y \mid x(a)) = \begin{cases} \prod_{t=1}^{T} p_{x(a),t}(y_t \mid y_{<t}), & \text{autoregressive,} \\ \prod_{t=1}^{T} K_t(y_t \mid x_t(a)), & \text{token replacement,} \end{cases}$$

It then predicts:

$$\hat{a} = \arg\max_{a \in \mathcal{A}} P(y \mid x(a)).$$

For a shuffled batch $Z = \{y_1, \ldots, y_n\}$, the attacker evaluates each prompt as the possible target, computes the corresponding batch likelihood, and averages over all n possible target prompts. The attribute value with the largest resulting likelihood is selected. The attack succeeds if the predicted attribute matches the ground-truth sensitive attribute.

C.1.7 Comparison with Existing Mechanism-aware Accountant

We compare our accountant with two recent mechanism-aware shuffle accountants Su et al. (2026); Takagi & Liew (2026). For Su et al. (2026), we explicitly construct the sequence-level distributions $P_x(y_{1:T})$, derive the privacy-blanket decomposition and privacy-amplification random variable, and apply their FFT-based accountant. For Takagi & Liew (2026), we similarly construct the sequence distributions and privacy blanket:

$$R_{\mathrm{BG}}(y) = \frac{\min_x P_x(y)}{\sum_{y'} \min_x P_x(y')},$$

and evaluate the finite-n blanket divergence using FFT.

C.1.8 Side-Channel Mitigation

In addition to protecting prompt contents, the deployment should prevent response length and timing from revealing the correspondence between users and shuffled requests. To mitigate these side channels, the service provider pads all responses within each group to a common length using masked values before returning them to the shuffler. The padded responses are then transmitted at a fixed rate of 20 tokens/s, preventing the shuffler or an external observer from distinguishing requests based on generation speed or individual response length.

C.2 Size of Context Group

For DP-Fusion and CusText, we summarize the number of genuine requests in each context group in Table 2. For DP-Fusion and CusText, privacy amplification is performed within context groups. We therefore examine the empirical distribution of group sizes in Table 2. The median group sizes are 7 and 8 for MMLU and MedicalQA, respectively, with sizes ranging from 1 to 28 for MMLU and from 1 to 20 for MedicalQA, indicating variation in the anonymity set across requests.

Table 2: **Distribution of context group sizes.** We report the number of context groups and the of the group size $n(c)$ over 5,000 genuine requests.

Dataset	# Groups	Size of context group $n(c)$					
		Min.	p_5	p_{50}	p_{95}	Max.	Mean
MMLU	636	1	3	7	15	28	7.86
MedicalQA	616	1	4	8	13	20	8.11

C.3 Latency Analysis

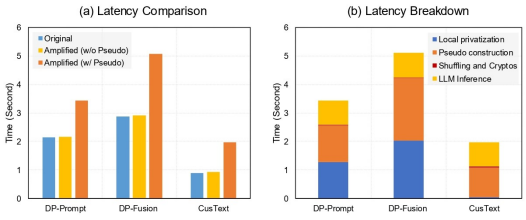


Figure 6: **Latency analysis on Gemini-3.1-flash.** (a) Latency comparison across original and amplified settings. (b) Latency breakdown for the amplified setting with pseudo prompt construction.

Latency consists of two components: an idle waiting period for collecting prompts from multiple clients and an active processing period for handling the assembled batch. The waiting time is determined by the configured collection window. We therefore focus on active processing latency, including: (i) service-side LLM inference; (ii) user-side local privatization; (iii) pseudo-prompt generation; and (iv) other overhead, primarily cryptographic operations and shuffling.

Figure 6(a) compares the original mechanisms with our framework, with and without pseudo prompts. Without pseudo-prompt construction, our framework incurs little additional latency, with an increase of less than 4%. Enabling pseudo prompts introduces additional overhead, primarily from pseudo-prompt construction, which accounts for more than 35% of the active processing time, as is shown in Figure 6(b). Nevertheless, the resulting end-to-end active processing latency remains below 5.2 seconds across all evaluated mechanisms.

C.4 Attack Results

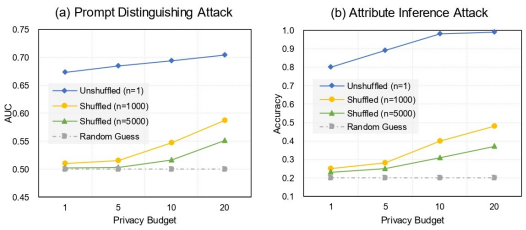


Figure 7: **Attack performance on the MMLU dataset.** We measure the attack performance by AUC for the prompt distinguishing attack and accuracy for the attribute inference attack. Shuffled ($n = 1000$) contains only genuine requests, and Shuffled ($n = 5000$) contains 1000 genuine requests and 4000 pseudo requests.

We consider two types of attacks to evaluate the privacy empirically: prompt distinguishing attack and attribute inference attack. In prompt distinguishing attack, the adversary compares the likelihood of the observed shuffled batch under two neighboring prompts and predicts which prompt generated it. In attribute inference attack, the adversary infers the most likely sensitive attribute from the privatized output.

As shown in Figure 7, under the same local privacy budget, shuffling substantially reduces the effectiveness of both attacks. Compared with the unshuffled setting, our shuffled framework reduces the AUC of prompt distinguishing attack by more than 15% and the accuracy of attribute inference attack by more than 25%. Adding pseudo requests to the batch further improves privacy protection, reducing the two attack metrics by up to 5% and 11%, respectively.

D Discussion

D.1 Multi-round Communication

To protect an entire conversation, adjacency changes one user's protected inputs across rounds while fixing other users' records. With at most one genuine report per user per batch, the accountant applies conditionally on prior transcripts visible to the service provider.

For R rounds, adaptive Rényi composition gives $L_\alpha^{\text{transcript}} \le \Lambda_\alpha = \sum_{r=1}^{R} \overline{L}_{\alpha,r}^{\text{shuf}}$ Mironov (2017). Let $\Gamma = \prod_{r=1}^{R} \gamma_r$ denote the product of uniform conditional overlap bounds, define the hockey-stick divergence as:

$$\delta_{\text{tot}}(\epsilon) = \int_{e^\epsilon}^{\Gamma^{-1}} \min\left\{ s^{-1}, \min_{\alpha \in \mathcal{A}} e^{\Lambda_\alpha} s^{-\alpha} \right\} ds.$$

Then the transcript is $(\epsilon_{\text{tot}}(\delta), \delta)$-DP, where

$$\epsilon_{\text{tot}}(\delta) = \inf\left\{ \epsilon \in [0, \log(\Gamma^{-1})] : \delta_{\text{tot}}(\epsilon) \le \delta \right\}.$$

In deployment, later requests would include previous conversation turns to preserve context. Growing histories increase privatization cost and can worsen the trade-off between privacy and utility. Locally generated summaries can limit this growth, with protected content privatized before transmission.

D.2 Token Length Variation

Prompts within a batch may have different output lengths, and privacy accounting is performed using the corresponding number of privatization steps. If token length is public, requests can be grouped by compatible lengths and accounted for separately, with the overall guarantee determined by the worst-case group. If length is private, prompts can instead be padded to a fixed public maximum length.

Very long prompts can weaken overlap bounds and increase the privacy budget. In practice, users may locally summarize them or decompose them into shorter subquestions before privatization. If one private prompt is split into multiple requests, these requests must be treated as contributions from the same user and their privacy losses composed, rather than counted as independent users in the shuffle.

D.3 Varying DP Mechanisms

Our privacy analysis assumes a common local privatizer. In practice, different mechanisms or privatization models may induce distinguishable output distributions, so equal local privacy budgets alone do not justify sharing a common overlap coefficient or anonymity set.

A direct extension is to partition requests by public privatization configuration. For each group g, we compute group-specific privacy parameters and generate pseudo reports using the same configuration. With fixed group membership and adjacency affecting one genuine request, parallel composition gives $\epsilon(\delta) = \max_g \epsilon_g(\delta)$. To reduce group fragmentation, the system can support a small set of standardized privatization configurations.