PRIVBLENDER: EFFICIENT DIFFERENTIALLY PRIVATE PROMPT PERTURBATION VIA BUDGET MIXING

Anonymous authors
Paper under double-blind review

ABSTRACT

The rapid development of large language models (LLMs) has heightened privacy risks associated with prompts containing sensitive information. Differentially private prompt perturbation has emerged as a promising approach to addressing these concerns, but existing methods incur substantial online computation overhead because of their reliance on the exponential mechanism. We present PrivBlender, an efficient offline–online framework that supports arbitrary target privacy budgets within a prescribed range, through presampling a small number of candidate outputs at different privacy budgets offline and selecting among them in constant time online. The key theoretical contribution is a novel privacy amplification result for mixtures of data-dependent exponential mechanisms: by appropriately mixing outputs generated at two privacy budgets ε_1 and ε_2, PrivBlender can achieve any target privacy budget $\varepsilon \in [\varepsilon_1, \varepsilon_2]$ and keep the cumulative privacy loss bounded by $\varepsilon_1 + \varepsilon_2$, regardless of the number of online releases. Experiments across multiple prompt-perturbation methods and downstream tasks show that PrivBlender substantially reduces online computation while matching or improving the utility of existing methods under the same formal privacy guarantee.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities across a wide range of natural language processing (NLP) tasks (Bubeck et al., 2023; Touvron et al., 2023; Gemini, 2025; OpenAI, 2026). However, adapting LLMs to domain-specific applications often requires exposing them to sensitive user or organizational data, raising severe privacy concerns. In particular, prompts submitted to LLMs may contain sensitive personal or proprietary information that can be directly revealed through model outputs or indirectly memorized and subsequently leaked by models fine-tuned on such inputs.

Under the umbrella of privacy-preserving NLP, a large body of work has adopted differential privacy (DP) (Dwork et al., 2006b) to address these privacy concerns. A particularly promising line of research (Yue et al., 2021; Chen et al., 2023; Tong et al., 2025; Wu et al., 2025) studies text sanitization through prompt perturbation. In this framework, an input prompt is first tokenized into a sequence of tokens. Each token is then perturbed using the exponential mechanism (McSherry & Talwar, 2007) to satisfy differential privacy before the resulting prompt is passed to the LLM. By the post-processing property of differential privacy, any subsequent computation on the perturbed prompt incurs no additional privacy loss.

However, as will be detailed in Section 2, DP prompt perturbation mechanisms suffer from an efficiency bottleneck inherently due to their reliance on the exponential mechanism. The perturbed token y for input token x can be any element from the vocabulary $\mathcal{V}$; consequently, perturbing each token requires evaluating $|\mathcal{V}|$ probabilities, followed by a weighted sampling procedure. A natural idea is to shift part of this contribution to an offline (i.e., pre-processing) stage, but doing so efficiently presents several challenges: (i) The sampling probabilities of the exponential mechanism depend on the target privacy parameter ε, which is typically specified *online* together with the user's prompt; (ii) One could prepare multiple perturbed prompts offline for a range of possible privacy budgets (e.g., $\varepsilon = 1, 2, 4, 8, \dots$) and select an appropriate one at query time. However, generating multiple releases from the same sensitive prompt incurs cumulative privacy loss under the composition property of differential privacy; (iii) At query time, it is only possible to use an offline release

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.

generated with $\varepsilon' \leq \varepsilon$. If the requested privacy budget ε does not match any precomputed value, the mechanism must use a smaller ε', resulting in unnecessary utility loss. Due to the above challenges, existing work only precomputes the $O(|\mathcal{V}|^2)$ size pairwise distance matrix during pre-processing, while the sampling procedure is still computed online, requiring $O(|\mathcal{V}|)$ computation per input token.

In this work, we present PrivBlender, an effective solution that addresses the above challenges. PrivBlender prepares a small number of perturbed tokens for each input token, using different privacy budgets during offline preprocessing, which carefully controls the cumulative privacy loss. At the online stage, PrivBlender selects a candidate token from this precomputed pool according to a carefully calibrated sampling probability, reducing the perturbation time of each token from $O(|\mathcal{V}|)$ to $O(1)$. Our analysis shows that any target privacy budget $\varepsilon \in [\varepsilon_1, \varepsilon_2]$ can be achieved by appropriately mixing tokens perturbed with privacy budgets ε_1 and ε_2, respectively. In particular, we make the following contributions:

1. **An efficient offline-online framework for DP prompt perturbation.** PrivBlender can be readily applied to existing prompt perturbation mechanisms, significantly improving their online efficiency while providing comparable or better utility under the same DP guarantee.
2. **An extension to privacy amplification tailored to the exponential mechanism.** As a key building block, our theoretical analysis establishes privacy amplification from mixing outputs of the exponential mechanism under different privacy budgets, which is of independent interest. This result not only improves upon existing results on privacy amplification by subsampling, but also provides a more general amplification framework that allows mixing with another data-dependent DP output.
3. **Comprehensive evaluation of efficiency and utility under unified metrics.** We implement PrivBlender on representative DP prompt perturbation mechanisms and evaluate it across diverse datasets, tasks, and privacy measures. In addition to a unified comparison under the standard DP guarantee, we also provide empirical privacy measures based on KNN and inference attacks.

The rest of the paper is organized as follows. Section 2 reviews relevant background on differential privacy, the exponential mechanism, and prompt perturbation. Section 3 presents our main theoretical result on privacy amplification, whereas Section 4 explains how PrivBlender works with existing prompt perturbation mechanisms. Empirical evaluation is provided in Section 5.

2 PRELIMINARIES

We denote by $\mathcal{V}$ the vocabulary, i.e., the set of all possible tokens. Following prior work, $\mathcal{V}$ is assumed public, which can be obtained from the language model's own tokenizer, or from predefined external word embeddings. An input prompt is tokenized into a private sequence of tokens[1] $(t_1, t_2, \ldots, t_n)$ where $t_i \in \mathcal{V}$ that can be fed into a large language model. The goal of *prompt perturbation* is to efficiently replace each token t_i with a perturbed token y_i under differential privacy, while preserving the utility of the original input.

2.1 DIFFERENTIAL PRIVACY

Differential privacy (DP) (Dwork & Nissim, 2004; Dwork et al., 2006a;b) is a rigorous framework for protecting individuals' private inputs from being inferred by an adversary, defined as follows.

Definition 1 (Differential Privacy (Dwork & Roth, 2014))**.** *Given a privacy budget $\varepsilon \geq 0$, a randomized mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ satisfies ε-DP if for any pair of neighboring inputs $x, x' \in \mathcal{X}$, and any possible output $y \in \mathcal{Y}$, it holds that*

$$\frac{\Pr[\mathcal{M}(x) = y]}{\Pr[\mathcal{M}(x') = y]} \leq e^{\varepsilon}\,. \tag{1}$$

Under the prompt perturbation setting, both the input space $\mathcal{X}$ and the output space $\mathcal{Y}$ are instantiated as the vocabulary $\mathcal{V}$. In this paper, we adopt token-level DP following the convention (Yue et al.,

[1] We refer to a token and its vector representation (word embedding) interchangeably in this paper.

2021; Chen et al., 2023; Tong et al., 2025; Wu et al., 2025), where any pair of input tokens $x, x' \in \mathcal{V}$ are considered neighbors, denoted $x \sim x'$. Intuitively, a DP mechanism assigns any pair of input tokens x, x' to the same output token y with indistinguishable probabilities, thus concealing which token was the true input.

2.2 Exponential Mechanism

The Exponential Mechanism (EM) is useful for releasing discrete non-numerical outputs under differential privacy while preserving the utility of the output measured by a utility function $u(x, y)$. This aligns with the target of prompt perturbation and has been widely adopted by existing work.

Definition 2 (Exponential Mechanism (EM) (McSherry & Talwar, 2007))**.** *Given a utility function $u : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that scores how desirable an output $y \in \mathcal{Y}$ is for a given input $x \in \mathcal{X}$, let*

$$\Delta := \max_{y \in \mathcal{Y}} \max_{x \sim x' \in \mathcal{X}} |u(x, y) - u(x', y)| \tag{2}$$

denote the sensitivity of u. The exponential mechanism $\mathcal{M}_{\mathrm{EM}} : \mathcal{X} \to \mathcal{Y}$ satisfies ε-differential privacy that samples an output $y \in \mathcal{Y}$ with probability

$$\Pr[\mathcal{M}_{\mathrm{EM}}(x) = y] = \exp\left(\frac{\varepsilon \cdot u(x, y)}{2\Delta}\right) \Bigg/ \sum_{y' \in \mathcal{Y}} \exp\left(\frac{\varepsilon \cdot u(x, y')}{2\Delta}\right) . \tag{3}$$

For prompt perturbation mechanisms, the utility function $u(x, y)$ can be designed to reflect semantic or task-specific similarity between the input token x and candidate output token y. For example, a common choice of $u(x, y) := -d_{\mathrm{euc}}(x - y)$ measures the Euclidean distance between the input and output word embeddings while prioritizing outputs that are similar to the input.

As we consider identical input and output domains, a special case of the Exponential Mechanism assigns $u(x, x) = 0$ for releasing the unperturbed input, and $u(x, y) = -\Delta$ for releasing any other output $y \neq x$. In this case, the Exponential Mechanism performs similarly to Generalized Randomized Response (Wang et al., 2016; 2017a;b), as it releases the input with probability $\frac{\exp(\varepsilon/2)}{\exp(\varepsilon/2)+|\mathcal{V}|-1}$, or randomly any other output $y \neq x$ from $\mathcal{V}$ with probability $\frac{1}{\exp(\varepsilon/2)+|\mathcal{V}|-1}$.

A major limitation of the exponential mechanism is that releasing each output requires computing $|\mathcal{Y}|$ probabilities, and sampling from the probability distribution, which also takes $O(|\mathcal{Y}|)$ time. In the worst case, perturbing n sensitive words from the input sentence can lead to $O(n|\mathcal{V}|)$ time consumption, which can be prohibitive.

2.3 Prompt Perturbation under Differential Privacy

All prior work on differentially private prompt perturbation (Yue et al., 2021; Chen et al., 2023; Tong et al., 2025; Wu et al., 2025) generally follows a token-level perturbation pipeline, although the privacy definitions and candidate sets differ across methods. Given an input sequence, each token is replaced by a token sampled from the same input set using EM or its variants. Subsequent tasks, including training and inference, access only the perturbed sequence and incur no additional privacy loss due to the post-processing property of DP (Dwork & Roth, 2014).

A common choice adopted in prior prompt perturbation mechanisms is to design the utility $u(x, y) = f(d(x, y))$ in Exponential Mechanism as a function of the distance $d(x, y)$ between the input and output tokens, where d can be Euclidean distance, cosine similarity, or their normalized values. Since all the distances are symmetric, we naturally have $d(x, y) = d(y, x)$ for any pair of tokens $x, y \in \mathcal{V}$. As a result, $u(x, y) = u(y, x)$, meaning the utility functions are also symmetric. This is a crucial property for our observation that the probability of outputting any token $y \in \mathcal{V}$ can be bounded in such mechanisms, which is formalized in Lemma 1.

Lemma 1. *Let $\mathcal{R}^{\varepsilon}_{\mathrm{EM}} : \mathcal{V} \to \mathcal{V}$ be an exponential mechanism with commutative utility function, i.e. $u(x, y) = u(y, x)$ for any $x, y \in \mathcal{V}$, then the probability of outputting any y for any input x is bounded by*

$$e^{-\varepsilon/2}/|\mathcal{V}| \leq \Pr[\mathcal{R}^{\varepsilon}_{\mathrm{EM}}(x) = y] \leq e^{\varepsilon/2}/|\mathcal{V}| \,.$$

The proof is folklore, which we include in Appendix A.1. The intuition is that by symmetry of the utility function, any pair of *outputs* also have indistinguishable probability, similar to the constraint imposed by DP on *inputs*. Consequently, any probability can only shift from a uniform $1/|\mathcal{V}|$ by no more than a multiplicative $e^{\varepsilon/2}$ factor.

3 Privacy Amplification by Mixing EM Outputs

Given input token x, privacy parameters $0 \leq \varepsilon_1 < \varepsilon_2$, and two independent presampled output tokens $y_{1,2} = \mathcal{R}_{\mathrm{EM}}^{\varepsilon_{1,2}}(x)$, the target is to generate an output token y that satisfies ε-DP for some $\varepsilon \in [\varepsilon_1, \varepsilon_2)$ with the help of offline results. A naive way of achieving the desired privacy guarantee is to directly release $y_1 = \mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}(x)$, since an ε_1-DP output is automatically ε-DP for $\varepsilon \geq \varepsilon_1$. However, this can lead to a degredation in utility if there is a large gap between ε_1 and ε. While output y_2 perturbed using ε_2 may provide better utility, its stated privacy guarantee violates the target ε guarantee, so y_2 cannot be released directly.

When $\varepsilon_1 = 0$, there is a general solution for the problem named *privacy amplification by subsampling* (Balle et al., 2018). Specialized to our setting, its guarantee is that if a mechanism releases an ε_2-DP result with probability p, and a data-independent output with probability $1-p$, then its overall privacy guarantee is $\varepsilon = \ln(1+p(e^{\varepsilon_2}-1))$. Since this holds for arbitrary ε_2 mechanisms, it suffices to set $p = \frac{\exp(\varepsilon)-1}{\exp(\varepsilon_2)-1}$ to achieve the target guarantee. However, existing results on privacy amplification through subsampling only support distribution interpolation between DP and data-independent (i.e. $\varepsilon_1 = 0$) outputs (Balle et al., 2018; Mironov et al., 2019; Fang & Yi, 2024). It remains unclear whether DP outputs under different privacy guarantees can be mixed to allow a more accurate characterization of the privacy guarantee, other than the naive bound $(\varepsilon_1 + \varepsilon_2)$ arising from composition and post-processing properties.

A major contribution of this paper is the analysis for the privacy guarantee of a Bernoulli mixture of two exponential mechanisms instantiated using different privacy budgets, detailed in Theorem 1.

Theorem 1 (Main Theorem)**.** *Let $\mathcal{R}_{\mathrm{EM}}^{\varepsilon_{1,2}} : \mathcal{V} \to \mathcal{V}$ each be an exponential mechanism with commutative utility function, where $0 \leq \varepsilon_1 < \varepsilon_2$. Then the mechanism $\mathcal{M} : \mathcal{V} \to \mathcal{V}$ defined as*

$$\mathcal{M}(x) = \begin{cases} \mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}(x), & \textit{with probability } 1-p \\ \mathcal{R}_{\mathrm{EM}}^{\varepsilon_2}(x), & \textit{with probability } p \end{cases}$$

satisfies ε-DP for $\varepsilon_1 \leq \varepsilon \leq \varepsilon_2$ if

$$p = \frac{\exp(\varepsilon-\varepsilon_1)-1}{\exp(\varepsilon-\varepsilon_1)-1+\exp(\frac{\varepsilon_2-\varepsilon_1}{2})-\exp(\varepsilon-\frac{\varepsilon_1+\varepsilon_2}{2})}.$$

The proof and detailed interpretation of the Theorem are included in Appendix B. As a special case, when $\varepsilon_1 = 0$, the output probability of each token becomes the same, and the exponential mechanism $\mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}$ degenerates into outputting a uniformly sampled token form domain $\mathcal{V}$ that is data-independent. In this case, the constraint on p in Theorem 1 becomes

$$p = \frac{\exp(\varepsilon)-1}{\exp(\varepsilon)-1+\exp(\frac{\varepsilon_2}{2})-\exp(\varepsilon-\frac{\varepsilon_2}{2})} = \frac{\exp(\varepsilon)-1}{(\exp(\frac{\varepsilon_2}{2})-1)(\exp(\varepsilon-\frac{\varepsilon_2}{2})+1)}.$$

4 PrivBlender for Prompt Perturbation

The design of PrivBlender is straightforward. Using the idea of two-stage sampling, the service provider prepares the outputs of $\mathcal{R}_{\mathrm{EM}}^{\varepsilon_{1,2}}(x)$ in its offline precomputing stage independently, and utilizes Theorem 1 to perturb online user prompts for any target $\varepsilon \in [\varepsilon_1, \varepsilon_2]$. While we defer the empirical selection of $\varepsilon_{1,2}$ to Section 5.5, here we discuss about three possible scenarios:

- **The user's prompt is also unavailable in the offline stage.** The service provider can adopt a context-independent embedding method (e.g. Word2Vec or GloVe) and prepare candidate tokens for each possible token $x \in \mathcal{V}$ using $O(|\mathcal{V}|^2)$ time and $O(|\mathcal{V}|)$ space. In the online phase, each token can be perturbed in $O(1)$ time, significantly improving over the $O(|\mathcal{V}|)$ time-per-token cost of existing work.

- **The user trusts the service provider, and is willing to reveal the prompt in the offline phase, but demands the service provider to protect her data in outbound analyses.** The service provider only needs to prepare for tokens that appear in user's prompt, and guarantees a cumulative privacy loss of $(\varepsilon_1 + \varepsilon_2)$ independent of how many times the user's prompt is used in analyses involving different ε's. The efficiency boost is similar.
- **The user's trust is higher to the service provider, but lower to outbound analysts.** In this case, the service provider receives a perturbed token from the user in the offline stage, used as the prompt for ε_2. The other prompt with $\varepsilon_1 = 0$ is data-independent, thus can also be generated by the service provider. The service provider then uses them to support outbound analyses with $\varepsilon \in [0, \varepsilon_2]$, without increasing the cumulative loss of ε_2.

In the following, we describe how PrivBlender work with four representative prompt perturbation mechanisms: SanText (Yue et al., 2021), CusText (Chen et al., 2023), RanText (Tong et al., 2025), and CAPE (Wu et al., 2025), to demonstrate its seamlessly integration with existing designs.

PB-SanText. SanText (Yue et al., 2021), the utility function is simply $u(x, y) = -d(x, y)$. When perturbed using $\Pr[\mathcal{M}_{\text{SanText}}(x)] \propto \exp(-\varepsilon_0 \cdot d(x, y)/2)$, the mechanism satisfies $(\varepsilon_0 \cdot d_{\max})$-DP[2], where $d_{\max} = \Delta = \max_{x,y \in \mathcal{V}} d(x, y)$ is the maximum distance between two tokens in the vocabulary. Since SanText directly follows the framework for prompt perturbation that we consider, its combination with PrivBlender is folklore.

PB-CusText. CusText (Chen et al., 2023) involves an additional step prior to perturbation that partitions the vocabulary $\mathcal{V}$ into disjoint size-K groups $\{\mathcal{V}_1, \ldots, \mathcal{V}_{\lceil |\mathcal{V}|/K \rceil}\}$. It iteratively picks an arbitrary unmapped token $x \in \mathcal{V}$, finds K semantically closest tokens to x, and generates a K-to-K mapping among them, until all tokens are mapped. For input $x_i \in \mathcal{V}_j$, CusText runs the exponential mechanism instantiated with $\mathcal{X} = \mathcal{Y} = \mathcal{V}_j$ and normalized distances, thus achieving a DP guarantee with respect to token pairs within the same vocabulary group.

As this additional partitioning step is data independent, PrivBlender for CusText naturally adopts the same partitioning while taking effect in the perturbation step, providing the same privacy guarantees.

PB-RanText. The RanText algorithm (Tong et al., 2025) replaces the deterministic domain partitioning in CusText by a random procedure. For input token x, a random token $\hat{x}$ is computed by injecting noise to the embedding of x. Any token x' within distance bounded by $d(x, x') < d(x, \hat{x})$ is considered semantically similar to x, and included in the vocabulary group $\mathcal{V}_x$ for x. The privacy guarantee of RanText is established with respect to each instantiation of $\mathcal{V}_x$.

As Tong et al. (2025) acknowledged in their Definition 1, the privacy guarantee of RanText does not naturally translate to a standard DP definition, since each token x generates its own vocabulary group $\mathcal{V}_x$ based on random coins, which can be different for neighboring tokens. The proof of Theorem 2 of in the paper is essentially conditioned on realized vocabulary groups $\mathcal{V}_x$. Once this is fixed, the analysis of RanText is identical to that of CusText, so that our PrivBlender can be similarly applied. The remaining difference is that RanText uses a different scoring function $u(x, y) = 1 - d(x, y)$, where d takes a normalized Euclidean distance. This only differs from the utility function in CusText by a constant additive factor, which has no effect on sampling probabilities as they are proportional to the exponent of utilities.

PB-CAPE. In CAPE (Wu et al., 2025), the authors consider "contextual logits" in addition to token distances, thus defining the utility function[3] as

$$u(x_i, y) = L_i(y)^{\lambda_L} \cdot \exp(-\lambda_D \cdot d(x_i, y))$$

where $L_i(y)$ denotes the logits of output y provided by a client-side model, which depends on the model used, preceding tokens $(x_1, \ldots, x_{i-1})$, and possibly subsequent tokens $(x_{i+1}, \ldots, x_n)$; d is the normalized Euclidean distance, and λ_L, λ_D are constant factors. CAPE runs a bucketed exponential mechanism: given input x_i, it partitions outputs $y \in \mathcal{V}$ into N_b buckets according to their utility $u(x_i, y)$, where each bucket has an equal utility range. The output for input x is sampled by first sampling a bucket using exponential mechanism, followed by drawing a uniform random

[2]The same mechanism also satisfies ε_0-MLDP (Metric LDP), which has a less strict constraint that only requires $\Pr[\mathcal{M}(x) = y] \leq e^{\varepsilon_0 \cdot d(x,x')} \cdot \Pr[\mathcal{M}(x') = y]$ for neighboring tokens $x \sim x'$ and output y. In this paper, we focus on the standard DP guarantee.

[3]As the utility function is non-symmetric, we provide a separate analysis in Appendix A.2.

element from the bucket. Its final privacy guarantee is $(\varepsilon + \varepsilon')$-DP, where ε naturally comes from the usage of EM, and ε' accounts for the worst-case difference in bucket sizes.

Similar to how PrivBlender integrates with SanText, CusText and RanText, combination of PrivBlender and CAPE applies to the EM sampling step, i.e., choosing the bucket for output y. In the offline phase, we prepare perturbed tokens by sampling the bucket using different privacy budgets, followed by sampling a uniform random item from the bucket. Theorem 1 applies to the first step, whereas the overhead in privacy for the second step is always ε'. In other words, PrivBlender mixes $(\varepsilon_1 + \varepsilon')$-DP and $(\varepsilon_2 + \varepsilon')$-DP outputs for the target $(\varepsilon + \varepsilon')$-DP when combined with CAPE. We provide a corresponding analysis in Appendix A.2.

5 Experiments

5.1 Experimental Setup

Datasets and Tasks. Following the setup of prior work, we consider two tasks in this paper: *binary classification* and *text generation*. For classification, we experiment on two datasets from the GLUE benchmark (Wang et al., 2019): **SST-2** contains sentiment annotations for movie reviews, and the task is to classify the sentiment as positive or negative; **QNLI** contains sentence pairs for entailment classification. For open-ended text generation, we use **Wikitext-103-v1** (Merity et al., 2017), a large-scale dataset derived from Wikipedia articles for language modeling tasks. Prompt configurations are included in Appendix C.

Utility and Efficiency. For classification, the reported utility is the classification accurcy, i.e. the fraction of prompts classified correctly. For open-ended text generation, utility is measured by the coherence and alignment of the perturbed prompts, following CAPE (Wu et al., 2025). Efficiency is measured by wall-clock time and peak memory consumption. We separate the cost of offline/online stages, where the offline stage of prior work involves computing the pairwise distance matrix, while PrivBlender also moves probability computation and sampling to its offline stage.

Privacy Measure. In addition to comparing all mechanisms under the same theoretical ε-DP guarantee. We also adopt empirical privacy measures through two privacy attacks: In **KNN Attack** (Song & Raghunathan, 2020), the adversary computes the distance between a perturbed token and all tokens in the vocabulary, selecting the k nearest neighbors as candidates; an attack succeeds if the original token is once of the candidates. In **Mask Token Inference (MTI) Attack** (Yue et al., 2021), the adversary uses a pre-trained BERT model to sequentially predict the masked token given the perturbed context and obtains a recovered sentence. The ROUGE-L F1 score is computed between the original and recovered sentence as the privacy leakage level. The privacy score is defined as $1 - r$, where r refers to prementioned attack success rates.

Mechanisms. We evaluate three mechanisms: **SanText**, **CusText** and **CAPE**, and our PrivBlender (**PB**) in combination with each of them as described in Section 4. We skip RanText since it degenerates to CusText when providing a formal DP guarantee using fixed partitioning. For CAPE, the privacy parameter ε presented is that of the exponential mechanism, as ε' is identical across all mechanisms. Details about the vocabulary and embedding configurations are included in Appendix D. Unless otherwise specified, ε_1 for our PrivBlender mechanism is fixed at 0. Detailed parameter choices are discussed in Section 5.5.

All experiments are conducted on a Linux server equipped with 2 Intel Xeon (2.70 GHz) CPUs, 881 GB RAM, and 6 NVIDIA H800 GPUs (80 GB each). Each result is an average of 10 trials.

5.2 Evaluation on Efficiency

Table 1 details the time and space consumption of PrivBlender in comparison with prior work. In the offline stage, PrivBlender typically incurs higher time and space consumption since it involves an additional step that computes EM probabilities and samples from them, respectively for ε_1 and ε_2. Naturally, the offline time cost of PrivBlender is roughly the corresponding base mechanism's offline cost plus twice its online cost. It is possible for common computations to be reused for the two executions in PrivBlender, for example, logits in CAPE mechanism.

Table 1: Comparison on Efficiency

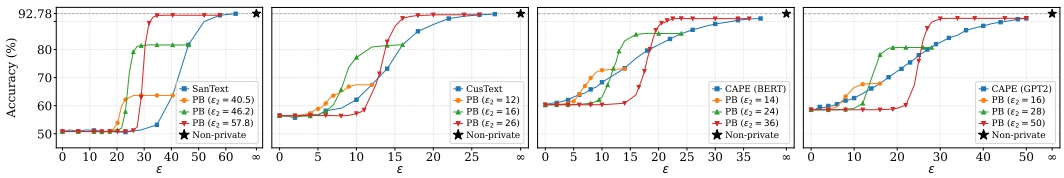
Mechanism	$\lvert\mathcal{V}\rvert$	Offline		Online	
		Time (s)	Memory (GB)	Time (s)	Memory (GB)
SanText	30,522	15.12	7.94	8.49	7.45
PB-SanText		30.02	7.94	0.01	0.04
CusText	65,713	57.86	1.34	0.70	0.47
PB-CusText		59.50	1.42	0.02	0.10
CAPE (BERT)	30,522	26.72	7.87	17.77	4.64
PB-CAPE (BERT)		53.84	7.88	0.04	0.61
CAPE (GPT2)	50,257	41.82	20.13	27.69	10.76
PB-CAPE (GPT2)		75.50	20.13	0.05	0.62

In its online stage, PrivBlender significantly improves over the corresponding base mechanism by as large as 500x in time and 180x in space. This is because for each input token, PrivBlender only maintains two perturbed tokens in its online stage, and performs a Bernoulli sampling in $O(1)$ time to randomly output one of them while satisfying the target privacy parameter ε. But for existing mechanisms, the computation of probabilities and token sampling in EM all happens during the online phase, which results in an $O(|\mathcal{V}|)$ cost in both time and space for each input token.

5.3 Evaluation on Utility

5.3.1 Classification Task

Following CAPE (Wu et al., 2025), we use a pretrained Qwen2-1.5B-Instruct model (Yang et al., 2024) for zero-shot classification on the validation sets of SST-2 and QNLI. We report the classification accuracy for all mechanisms varying the target privacy guarantee ε. For PrivBlender, we fix $\varepsilon_1 = 0$, and present accuracies achieved through mixing with different values of ε_2.

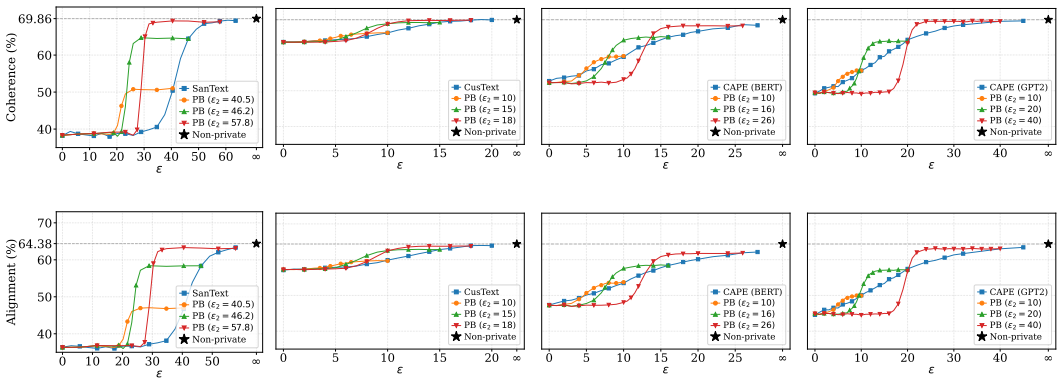


(a) SanText on SST-2 (b) CusText on SST-2 (c) CAPE (BERT) on SST-2 (d) CAPE (GPT2) on SST-2

Figure 1: Utility on SST-2 varying the privacy guarantee ε.

We present results on SST-2 in Figure 1 and defer similar results on QNLI to Appendix E.1. The non-private baseline on SST-2 achieves accuracy 92.78%, denoted by a star in each subplot. Since the task is a binary classification, a random token ($\varepsilon = 0$) achieves 50% accuracy, as shown in Figure 1a for SanText near $\varepsilon = 0$. The accuracies of CusText and CAPE are above 50% even when $\varepsilon = 0$, because the final privacy guarantee for CusText is only with respect to token pairs within the same group, and that for CAPE is $\varepsilon' + \varepsilon$.

For PrivBlender, its performance naturally follows the sampling probability plotted in Figure 6. In particular, its accuracy is close to that of ε_1 or ε_2 when the target ε is near to either endpoint, while an arithmetic mean of the accuracies from ε_1 and ε_2 is almost achieved for $\varepsilon = (\varepsilon_1 + \varepsilon_2)/2$. Since the sampling probability is a concave function of ε when $\varepsilon > (\varepsilon_1 + \varepsilon_2)/2$, PrivBlender can even dominate the utility of the baseline mechanism under the same ε-DP guarantee. Recall that the primary focus of PrivBlender is to improve online efficiency, as has been evaluated in Section 5.2. The improvement on accuracy over baseline comes at no additional cost for large ε, in addition to the efficiency improvement enjoyed by ε in the whole range $[\varepsilon_1, \varepsilon_2]$.



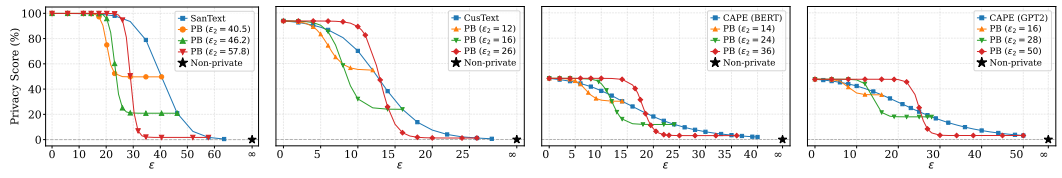
(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

Figure 2: Coherence and Alignment on Wikitext varying the privacy guarantee ε.

5.3.2 Text Generation Task

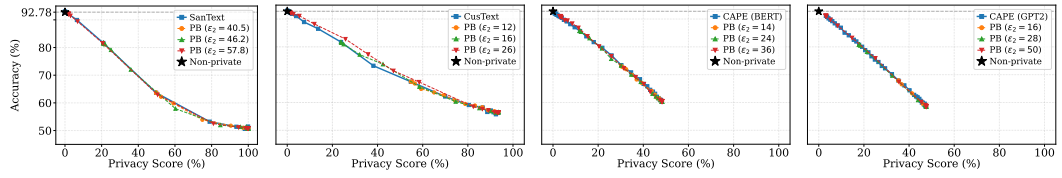
We present the coherence and alignment for all mechanisms in Figure 2. Similarly, we fix $\varepsilon_1 = 0$ for PrivBlender and consider different ε_2's. The general performance similarly follows the S-shaped sampling probability between ε_1 and ε_2, as discussed earlier. For SanText, as the baseline mechanism does not significantly out-perform the data-independent execution for $\varepsilon < 30$, PrivBlender strictly dominates the baseline since it is not penalized for releasing a token for ε_1, but rewarded for sampling a token for ε_2. For CusText and CAPE, it is possible for PrivBlender to have less utility if ε_2 is too large for the target ε. However, recall that PrivBlender significantly saves the only computation cost, and this can be avoided by adopting an appropriate offline candidate with $\varepsilon_2 \leq 2\varepsilon$, as will be discussed in Section 5.5.

5.4 Evaluation on Privacy



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT-2)

Figure 3: Privacy score under KNN attack on SST-2.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT-2)

Figure 4: Privacy-utility tradeoff under KNN attack on SST-2

Figure 3 presents the token-level privacy score under KNN attack (Song & Raghunathan, 2020). In general, larger ε corresponds to lower privacy scores. Due to the difference in their privacy notion, CusText and CAPE do not achieve perfect privacy even when $\varepsilon = 0$ in the Figures. Despite the same nominal DP guarantee, the empirical privacy score of PrivBlender is also a mixture of that from the candidates it mixes, following the sampling probability. The privacy-utility tradeoff measured by KNN attack score is presented in Figure 4, and results under the MTI attack and

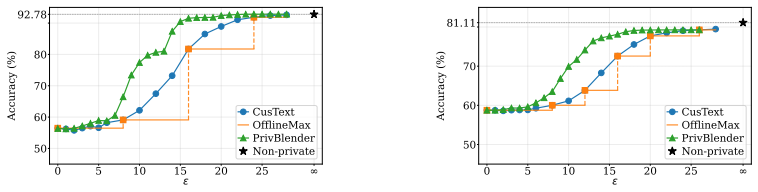
those on the QNLI dataset can be found in Appendix E.2. It is clear that the empirical trade-off is similar, but PrivBlender is stronger in providing a tighter worst-case nominal privacy guarantee under differential privacy.

5.5 Offline Candidate Selection

As indicated by Figure 4, the empirical privacy-utility trade-off is similar for different instantiations of PrivBlender when measured under KNN attack. The difference is essentially in the "nominal" DP guarantee formally achieved by mixing ε_1 and ε_2. Given that the KNN attack is task-agnostic, we can optimize for a mixture that *minimizes* the empirical privacy score measured under KNN attack, which correspondingly *maximizes* the utility, while providing the same nominal privacy guarantee.

Figure 14 in Appendix E.3 illustrates this process. Assume that the service provider is preparing offline candidates for an expected target privacy budget $\varepsilon = 14$. She searches among $\varepsilon_1 \in [0, \varepsilon]$ and $\varepsilon_2 \in [\varepsilon, 2\varepsilon]$, with the objective of minimizing the empirical KNN privacy score. Note that for fixed ε_1 and ε_2, the privacy score is a weighted average of those from ε_1 and ε_2, which can also be computed in constant time. Figure 14a plots the privacy score heatmap for $\varepsilon_{1,2}$ pairs, where the minimum is found at $(\varepsilon_1, \varepsilon_2) = (0, 22)$ with sampling probability is $p = 0.953$. There is a matching maximum for utility at 89.49%. Compared to directly running the baseline mechanism with $\varepsilon = 14$ whose utility is 73.2%, PrivBlender achieves the same nominal DP guarantee but improved utility. Equivalently, the DP guarantee for PrivBlender is tighter when considering empirical loss.

While PrivBlender supports any target ε within the prescribed range, it also allows the service provider to prepare additional intermediate candidates for better performance. Figure 5 illustrates the idea. Suppose the service provider prepares $\varepsilon_{\text{offline}} = \{0, 8, 16, 24\}$ for SST-2, then a natural baseline is to use the maximum offline candidate ε' where $\varepsilon' \leq \varepsilon$ for the prompt. This is denoted by OfflineMax in Figure 5a. With PrivBlender, the service provider can optimize for the best pair of $\varepsilon_{1,2} \in \varepsilon_{\text{offline}}$ to mix for each target ε, and achieve a smooth variation of utility that strictly dominates the baseline solution.



(a) SST-2, $\varepsilon_{\text{offline}} = \{0, 8, 16, 24\}$ (b) QNLI, $\varepsilon_{\text{offline}} = \{0, 8, 12, 16, 20, 24\}$

Figure 5: Utility of PrivBlender with multiple offline candidates using CusText as the baseline.

6 Conclusion

In this work, we present PrivBlender, an efficient framework that prepares offline candidate perturbed tokens using existing mechanisms that are based on the Exponential Mechanism. In its online stage, PrivBlender is capable of mixing its offline candidates to satisfy the target privacy budget ε. This is utilized by our novel theoretical finding for Bernoulli mixture of independent EM outputs in Theorem 1, which generalizes and improves over privacy amplification through subsampling and is of independent interest. Our experiments have demonstrated that the algorithm not only is favorable for its online efficiency, but can also improve the utility compared to direct running baseline mechanisms under the target privacy budget, as it provides a tighter nominal privacy measure that leads to lower empirical privacy scores and therefore higher utility. We have also demonstrated how offline candidates can be selected, and how our mechanism supports multiple offline candidates.

As a future direction, our current formulation of Theorem 1 assumes a commutative utility function in EM. While we can also prove similar results for specific functions that are not commutative (e.g. for CAPE in Appendix A.2), it will be interesting if it can be extended to allow mixing general DP outputs instead of only those from EMs.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods and editing the paper to improve readability. We have reviewed all AI-assisted work. For mechanism implementations, LLM generated code was verified and tested for correctness by 3 of the authors. In addition, we utilize the theoretical guarantee of our mechanism to ensure the correctness in implementation. AI assisted in grammar and typo checking, where each edit advice was performed manually by an author.

We have not used generative AI tools for formulating mathematical claims, providing critical ingredients for proving mathematical claims, assistance in the writing of proofs, designing or providing feedback on research methodology or experiments, supporting qualitative and thematic data analysis, interpreting results, suggesting experimental parameters, drafting the paper, summarizing or analysing existing literature, discovering research topics or identifying gaps, brainstorming, sourcing/searching for information, identifying relevant literature, formatting references, suggesting a structure for the paper, nor proposing a title or keywords for the research paper. The core research idea, including the mechanisms designs and theoretical contributions (with its proofs) was developed entirely by the authors without AI assistance.

Generation of synthetic data sets, developing theoretical models or conceptual frameworks, proposing or refining hypotheses, assistance with translation, cleaning and reformating dataset, formulating questions for surveys or interviews, creating or modifying scientific figures or images, creating or editing software code, creation of artifacts, and transcribing recordings of research material are not applicable to this work.

ETHICS STATEMENT

This paper focuses on differential privacy, a widely-adopted framework for privacy protection. All the datasets used in experiments are public ones from existing work. There should be no ethics concerns as far as we are concerned.

REPRODUCIBILITY STATEMENT

The code is available at `https://anonymous.4open.science/r/PrivBlender/`. The repository contains three folders, each for a base mechanism (i.e. SanText, CusText, and CAPE) and its corresponding integration with PrivBlender. Commands for reproducing each experiment result is detailed in the README file. For example, under the `SanText` folder, `run_santext.sh` is the script for executing the original SanText mechanism, and `run_privblender.sh` is the script for our new PB-SanText mechanism. Both accept parameter ε, while for PrivBlender there are additional offline parameters ε_1 and ε_2. For details on reproducing each figure, please refer to the full README file.

REFERENCES

Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by subsampling: Tight analyses via couplings and divergences. In *NeurIPS*, pp. 6280–6290, 2018. URL https://dl.acm.org/doi/10.5555/3327345.3327525.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Túlio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712, 2023. doi: 10.48550/ARXIV.2303.12712. URL https://doi.org/10.48550/arXiv.2303.12712.

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 *ACL (Findings)*, volume ACL 2023 of *Findings of ACL*, pp. 5747–5758. Association for Computational Linguistics, 2023. URL https://aclanthology.org/2023.findings-acl.355/.

Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pp. 528–544. Springer, 2004. URL https://doi.org/10.1007/978-3-540-28628-8_32.

Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014. URL https://dl.acm.org/doi/10.1561/0400000042.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pp. 486–503. Springer, 2006a. URL https://doi.org/10.1007/11761679_29.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pp. 265–284. Springer, 2006b. URL https://doi.org/10.1007/11681878_14.

Juanru Fang and Ke Yi. Privacy amplification by sampling under user-level differential privacy. *Proc. ACM Manag. Data*, 2(1):34:1–34:26, 2024. doi: 10.1145/3639289. URL https://doi.org/10.1145/3639289.

Gemini. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261, 2025. doi: 10.48550/ARXIV.2507.06261. URL https://doi.org/10.48550/arXiv.2507.06261.

Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *FOCS*, pp. 94–103. IEEE Computer Society, 2007. URL https://doi.org/10.1109/FOCS.2007.66.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=Byj72udxe.

Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *CoRR*, abs/1908.10530, 2019. URL http://arxiv.org/abs/1908.10530.

OpenAI. Openai GPT-5 system card. *CoRR*, abs/2601.03267, 2026. doi: 10.48550/ARXIV.2601.03267. URL https://doi.org/10.48550/arXiv.2601.03267.

Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *CCS*, pp. 377–390. ACM, 2020. URL https://doi.org/10.1145/3372297.3417270.

Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi, Weiming Zhang, Nenghai Yu, Tianwei Zhang, and Zhikun Zhang. InferDPT: Privacy-preserving inference for closed-box large language models. *IEEE Trans. Dependable Secur. Comput.*, 22(5):4625–4640, 2025. URL https://doi.org/10.1109/TDSC.2025.3550389.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023. doi: 10.48550/ARXIV.2302.13971. URL https://doi.org/10.48550/arXiv.2302.13971.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=rJ4km2R5t7.

Shaowei Wang, Liusheng Huang, Pengzhan Wang, Hou Deng, Hongli Xu, and Wei Yang. Private weighted histogram aggregation in crowdsourcing. In Qing Yang, Wei Yu, and Yacine Challal (eds.), *Wireless Algorithms, Systems, and Applications - 11th International Conference, WASA 2016, Bozeman, MT, USA, August 8-10, 2016. Proceedings*, volume 9798 of *Lecture Notes in Computer Science*, pp. 250–261. Springer, 2016. doi: 10.1007/978-3-319-42836-9_23. URL https://doi.org/10.1007/978-3-319-42836-9_23.

Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In Engin Kirda and Thomas Ristenpart (eds.), *26th USENIX Security Symposium, USENIX Security 2017, Vancouver, BC, Canada, August 16-18, 2017*, pp. 729–745. USENIX Association, 2017a. URL https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/wang-tianhao.

Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Optimizing locally differentially private protocols. *CoRR*, abs/1705.04421, 2017b. URL http://arxiv.org/abs/1705.04421.

Haoqi Wu, Wei Dai, Li Wang, and Qiang Yan. Cape: Context-aware prompt perturbation mechanism with differential privacy. In *ICML*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025. URL https://openreview.net/forum?id=cLxLgpMd2v.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *CoRR*, abs/2407.10671, 2024. doi: 10.48550/ARXIV.2407.10671. URL https://doi.org/10.48550/arXiv.2407.10671.

Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. Differential privacy for text analytics via natural text sanitization. In *ACL/IJCNLP (Findings)*, volume ACL-IJCNLP 2021 of *Findings of ACL*, pp. 3853–3866. Association for Computational Linguistics, 2021. URL https://aclanthology.org/2021.findings-acl.337/.

A Proofs and Analyses

A.1 Proof of Lemma 1

Proof. Recall that $\Delta := \max_{y\in\mathcal{V}} \max_{x\sim x'\in\mathcal{V}} |u(x,y) - u(x',y)|$ is the sensitivity of the utility function. Since any pair of input tokens $x, x' \in \mathcal{V}$ are considered neighbors, we have $|u(x,y) - u(x',y)| \le \Delta$ for all $x, x', y \in \mathcal{V}$. By symmetry and renaming of symbols, $|u(x,y) - u(x,y')| \le \Delta$ is also true for all $x, y, y' \in \mathcal{V}$. The probability of outputting any y from any x can be bounded as

$$\begin{aligned}\Pr[\mathcal{M}(x) = y] &= \frac{\exp\left(\frac{\varepsilon \cdot u(x,y)}{2\Delta}\right)}{\sum_{y'\in\mathcal{Y}} \exp\left(\frac{\varepsilon \cdot u(x,y')}{2\Delta}\right)} \\ &= \left(1 + \sum_{y'\neq y} \exp\left(\frac{\varepsilon \cdot (u(x,y') - u(x,y))}{2\Delta}\right)\right)^{-1} \\ &\ge \left(1 + (|\mathcal{V}| - 1) \cdot e^{\varepsilon/2}\right)^{-1} \ge e^{-\varepsilon/2}/|\mathcal{V}|\,.\end{aligned}$$

The other direction follows from $\Pr[\mathcal{M}(x) = y] \le \left(1 + (|\mathcal{V}| - 1) \cdot e^{-\varepsilon/2}\right)^{-1} = \frac{e^{\varepsilon/2}}{e^{\varepsilon/2} + |\mathcal{V}| - 1}$. □

A.2 Detailed privacy Analysis of CAPE with PrivBlender

CAPE Mechanism Review. For an input sequence x, CAPE (Wu et al., 2025) scores output $y \in \mathcal{V}$ for the i-th token x_i with a hybrid function $h(x_i, y) = \ell_i(y)^{\lambda_\ell} \cdot e^{-d_{\text{norm}}(x_i,y)\cdot\lambda_d}$, where $\ell_i(y)$ denotes

the contextual logits of y predicted by a local model (BERT or GPT2), and $d_{\text{norm}}(x_i, y)$ is the normalized embedding distance. The importance factors are $\lambda_\ell = 0.5$ and $\lambda_d = 1$ by default. Given all scores, candidates are sorted and partitioned into equal-width buckets $B(x_i)$, with the score of a bucket $b \in B(x_i)$ defined as $u(x_i, b) := \frac{1}{|b|}\sum_{y\in b} h(x_i, y)$. CAPE then applies the exponential mechanism for selecting a bucket b for x_i with utility function u, followed by outputting a uniformly random $y \in b$ as output.

Denote the bucket for input x_i and output y as $b_{x_i,y}$. In the strictest sense, the sensitivity of the utility function $u(x_i, b)$ is

$$\Delta_u := \max_{x_i \sim x_i'} \max_{y\in\mathcal{V}} |u(x_i, b_{x_i,y}) - u(x_i', b_{x_i',y})| .$$

However, computing Δ_u in exact requires enumerating all neighboring x_i' and all y for all x_i in $O(|\mathcal{V}|^3)$ time, since the bucketing depends on the input token and the logits depend on the context. Provided that Wu et al. (2025) did not discuss about how the exact computation can be effectively done, we adopt an upper bound $\Delta \geq \Delta_u$ in our paper, defined as $\Delta := L^{\lambda_\ell}$ where $L \geq \ell_i(y)$ for all x_i, y is an upper bound on the logits returned by the local model[4]. We have $h(x_i, y) \in [0, \Delta]$ since the normalized distance $d_{\text{norm}}(x_i, y) \in [0, 1]$. As $u(x_i, b)$ computes the mean of $h(x_i, y)$ for $y \in b$, we naturally have $u(x_i, b) \in [0, \Delta]$ and the sensitivity bound follows. Our experiments show that such choice of Δ matches with the experiment results presented in the original paper.

It should be noted that our assumption on a commutative utility function in Lemma 1 no longer holds for utility function $u(x_i, b)$, as $x_i \in \mathcal{V}$ and $b \in B(x_i)$ are from different domains. Nevertheless, the conclusion of Lemma 1 is unaffected: that the probability of any output is bounded. Indeed, this is the only property we use in the proof of Theorem 1, and we prove it in the following.

The key observation is that the difference in utility between any pair of outputs $b, b' \in B(x_i)$ for input x_i is still bounded, as

$$\max_{x_i\in\mathcal{V}} \max_{b,b'\in B(x_i)} |u(x_i, b) - u(x_i, b')| \leq \max_{x_i\in\mathcal{V}} \max_{y,y'\in\mathcal{V}} |h(x_i, y) - h(x_i, y')| \leq \Delta ,$$

following the range of function h. Let N_b denote the number of buckets in $B(x_i)$, the probability that b is selected for output is

$$\begin{aligned}
\Pr[\mathcal{R}(x_i) = b] &= \frac{\exp(\frac{\varepsilon\cdot u(x_i,b)}{2\Delta})}{\sum_{b'\in B(x_i)} \exp(\frac{\varepsilon\cdot u(x_i,b')}{2\Delta})} \\
&= \left(1 + \sum_{b'\neq b} \exp\left(\frac{\varepsilon\cdot(u(x_i, b') - u(x_i, b))}{2\Delta}\right)\right)^{-1} \\
&\geq \left(1 + (N_b - 1)\cdot e^{\varepsilon/2}\right)^{-1} \geq e^{-\varepsilon/2}/N_b
\end{aligned}$$

The other direction is similar, thus we have

$$e^{-\varepsilon/2} \leq N_b \cdot \Pr[\mathcal{R}(x_i) = b] \leq e^{\varepsilon/2} . \tag{4}$$

PrivBlender with CAPE. For CAPE, the probability of outputting token y is then

$$\Pr[\mathcal{M}^{\varepsilon+\varepsilon'}_{\text{CAPE}}(x_i) = y] = \Pr[\mathcal{R}^{\varepsilon}(x_i) = b_{x_i,y}] \cdot \frac{1}{|b_{x_i,y}|} .$$

On any pair of neighboring datasets $x_i \sim x_i'$ and any output y, we can bound

$$\begin{aligned}
\frac{\Pr[\mathcal{M}(x_i) = y]}{\Pr[\mathcal{M}(x_i') = y]} &= \frac{(1-p)\Pr[\mathcal{M}^{\varepsilon_1+\varepsilon'}_{\text{CAPE}}(x_i) = y] + p\Pr[\mathcal{M}^{\varepsilon_2+\varepsilon'}_{\text{CAPE}}(x_i) = y]}{(1-p)\Pr[\mathcal{M}^{\varepsilon_1+\varepsilon'}_{\text{CAPE}}(x_i') = y] + p\Pr[\mathcal{M}^{\varepsilon_2+\varepsilon'}_{\text{CAPE}}(x_i') = y]} \\
&= \frac{(1-p)\Pr[\mathcal{R}^{\varepsilon_1}(x_i) = b_{x_i,y}] + p\Pr[\mathcal{R}^{\varepsilon_2}(x) = b_{x_i,y}]}{(1-p)\Pr[\mathcal{R}^{\varepsilon_1}(x_i') = b_{x_i',y}] + p\Pr[\mathcal{R}^{\varepsilon_2}(x_i') = b_{x_i',y}]} \cdot \frac{|b_{x_i',y}|}{|b_{x_i,y}|} \\
&\leq \frac{(1-p)\exp(\varepsilon_1/2) + p\exp(\varepsilon_2/2)}{(1-p)\exp(-\varepsilon_1/2) + p\exp(-\varepsilon_2/2)} \cdot \frac{|b_{x_i',y}|}{|b_{x_i,y}|} ,
\end{aligned}$$

[4]We have $L = 50$ for BERT and $L = 70$ for GPT2.

where the inequality is obtained by plugging in equation 4. Compared to the proof of Theorem 1, the additional multiplicative factor $\frac{|b_{x_i',y}|}{|b_{x_i,y}|}$ is bounded by a data-independent factor ε', where empirically $\varepsilon \approx 8.6$ for BERT and $\varepsilon' \approx 9.4$ for GPT2 in our setting. Correspondingly, the optimal choice of p is the same as in Theorem 1 as a function of ε_1 and ε_2, and PrivBlender essentially mixes $(\varepsilon_1+\varepsilon')$-DP and $(\varepsilon_2+\varepsilon')$-DP executions of the CAPE mechanism into a final output with $(\varepsilon+\varepsilon')$-DP guarantee, where ε is determined by p as in Theorem 1.

B Proof and Interpretation of Theorem 1

We first provide the proof for Theorem 1.

Proof. By expanding the probability distribution of $\mathcal{M}$, we have

$$\Pr[\mathcal{M}(x)=y] = (1-p)\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}(x)=y] + p\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_2}(x)=y]\,.$$

It follows that for any pair of $x, x', y \in \mathcal{V}$,

$$\begin{aligned}\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} &= \frac{(1-p)\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}(x)=y] + p\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_2}(x)=y]}{(1-p)\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_1}(x')=y] + p\cdot\Pr[\mathcal{R}_{\mathrm{EM}}^{\varepsilon_2}(x')=y]}\\ &\le \frac{(1-p)\cdot\frac{\exp(\varepsilon_1/2)}{|\mathcal{V}|} + p\cdot\frac{\exp(\varepsilon_2/2)}{|\mathcal{V}|}}{(1-p)\cdot\frac{\exp(-\varepsilon_1/2)}{|\mathcal{V}|} + p\cdot\frac{\exp(-\varepsilon_2/2)}{|\mathcal{V}|}}\\ &= \frac{\exp(\varepsilon_1) + p(\exp(\frac{\varepsilon_1+\varepsilon_2}{2}) - \exp(\varepsilon_1))}{1 + p(\exp(\frac{\varepsilon_1-\varepsilon_2}{2}) - 1)}\,,\end{aligned}$$

where the inequality is by plugging in bounds in Lemma 1. Solving for $\frac{\Pr[\mathcal{M}(x)=y]}{\Pr[\mathcal{M}(x')=y]} \le \exp(\varepsilon)$ gives

$$p \le \frac{\exp(\varepsilon)-\exp(\varepsilon_1)}{\exp(\varepsilon)-\exp(\varepsilon_1)+\exp(\frac{\varepsilon_1+\varepsilon_2}{2})-\exp(\varepsilon+\frac{\varepsilon_1-\varepsilon_2}{2})}\,,$$

which finishes the proof. □

Recall that in Theorem 1, p is the probability of releasing the token perturbed with ε_2, whose utility is better compared to the token perturbed with ε_1. Therefore, the final utility of the ε-DP output is positively related to p. Specially, when $\varepsilon = \varepsilon_1$, we have $p = 0$, meaning mechanism $\mathcal{M}$ always returns the token perturbed using ε_1; when $\varepsilon = \varepsilon_2$, we have $p = 1$, and $\mathcal{M}$ always returns the token perturbed using ε_2. When $\varepsilon = (\varepsilon_1+\varepsilon_2)/2$, it can be verified that $p = 1/2$, where $\mathcal{M}$ uniformly randomly chooses between the two tokens to achieve an amplified privacy guarantee in between. In addition, we can prove that p as a function of ε is S-shaped: it is convex on the first half, and concave on the second half.

Lemma 2. *For $0 < \varepsilon_1 < \varepsilon_2$, the function $p(\varepsilon)$ in Theorem 1 as a function of ε is strictly increasing on $[\varepsilon_1, \varepsilon_2]$, while being convex on $\left[\varepsilon_1, \frac{\varepsilon_1+\varepsilon_2}{2}\right]$ and concave on $\left[\frac{\varepsilon_1+\varepsilon_1}{2}, \varepsilon_2\right]$.*

Proof. We have

$$\begin{aligned}p(\varepsilon) &= \frac{\exp(\varepsilon-\varepsilon_1)-1}{\exp(\varepsilon-\varepsilon_1)-1+\exp(\frac{\varepsilon_2-\varepsilon_1}{2})-\exp(\varepsilon-\frac{\varepsilon_1+\varepsilon_2}{2})}\\ &= \frac{\exp(\varepsilon-\varepsilon_1)-1}{\left(\exp(\frac{\varepsilon_2-\varepsilon_1}{2})-1\right)\left(\exp(\varepsilon-\frac{\varepsilon_1+\varepsilon_2}{2})+1\right)}\end{aligned}$$

As $\exp(\frac{\varepsilon_2-\varepsilon_1}{2}) - 1 > 0$ is a constant, it suffices to analyze

$$q(\varepsilon) := \frac{\exp(\varepsilon-\varepsilon_1)-1}{\exp(\varepsilon-\frac{\varepsilon_1+\varepsilon_2}{2})+1}\,.$$

Denote by $A := \exp(\varepsilon-\varepsilon_1)$ and $B := \exp(\varepsilon-\frac{\varepsilon_1+\varepsilon_2}{2})$, the derivative of q is simply

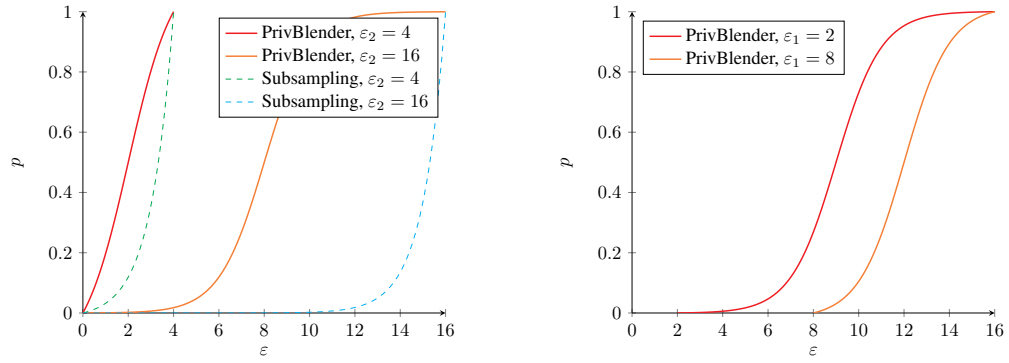
$$q'(\varepsilon) = \frac{(B+1)A-(A-1)B}{(B+1)^2} = \frac{A+B}{(B+1)^2} > 0\,,$$

therefore q (thus p) is strictly increasing. Its second-order derivative is

$$q''(\varepsilon) = \frac{(B+1)^2 \cdot (A+B) - (A+B) \cdot 2(B+1) \cdot B}{(B+1)^4} = \frac{(1-B)(A+B)}{(B+1)^3}.$$

Its sign is determined by the sign of $1 - \exp(\varepsilon - \frac{\varepsilon_1+\varepsilon_2}{2})$. When $\varepsilon < \frac{\varepsilon_1+\varepsilon_2}{2}$, we have $q''(\varepsilon) > 0$, therefore q (thus p) is convex. When $\varepsilon > \frac{\varepsilon_1+\varepsilon_2}{2}$, we have $q''(\varepsilon) < 0$ and q (thus p) is concave.

□



(a) Mixing $\varepsilon_1 = 0$ with ε_2

(b) Mixing ε_1 with $\varepsilon_2 = 16$

Figure 6: Relationship between probability p and target privacy ε

Figure 6 plots the relationship between the probability p of releasing the higher-utility result versus the target privacy ε. There are two advantages of the amplification in Theorem 1. First, since our analysis is tailored to exponential mechanism with symmetric utility functions, it provides a tighter guarantee than the generic privacy amplification by subsampling for arbitrary DP mechanisms, as shown in Figure 6a. For example, when mixing $\varepsilon_1 = 0$ and $\varepsilon_2 = 16$ for a target privacy guarantee $\varepsilon = 8$, our PrivBlender mechanism has $p = 1/2$ probability of choosing ε_2, the result with higher utility; whereas in privacy amplification through subsampling, this probability is $\frac{e^8-1}{e^{16}-1} \approx 0$, meaning the mechanism almost always releases an data-independent result. Second, PrivBlender also supports mixing two DP outputs from EM that are both data-dependent, whose privacy cannot be argued from existing results. As Figure 6b shows, through mixing $\varepsilon_1 = 2$ with $\varepsilon_2 = 16$, PrivBlender has the flexibility of adapting to an arbitrary target privacy guarantee $\varepsilon \in [2, 16]$.

C Prompt Templates

This section presents the prompt templates used in our experiments, including both the classification tasks and the text generation task.

C.1 Prompts for Binary Classification Tasks

Table 2: System prompts and user templates for binary classification tasks.

Task	System Prompt	User Template
SST-2	You are a helpful assistant. Classify if the sentence is positive or negative sentiment. Just gives the answer in positive, negative.	----"Content": {sentence}
QNLI	You are a helpful assistant trained to determine whether a given context sentence contains the information needed to answer the given question. If the context answers the question, respond "Yes". Otherwise, respond "No".	----"Question": {question} ----"Context": {sentence}

For the classification tasks (SST-2 and QNLI), we design task-specific system prompts and user templates to perform zero-shot classification. Table 2 summarizes the prompts used for each task.

C.2 Prompts for Text Generation Task

For the text generation task, our pipeline consists of two stages: (1) a *generation stage*, which simulates the remote server generating a continuation based on the (perturbed) prefix text, and (2) an *extraction stage*, performed locally, which extracts coherent and consistent content from the noisy generation to produce the final extended text. Table 3 presents the prompts used for each stage.

Table 3: System prompts and user templates for the two stages of the text generation task.

Stage	**System Prompt**	**User Template**
Generation (Server)	Your task is to extend "`Prefix Text`". Provide only your Continuation.	`----"Prefix Text"`: {prompt} `----"Continuation":`
Extraction (Local Model)	Your task is to extend the "`Prefix Text`". Use the "`Perturbed Generation`" as your primary writing material for your extension. Extract coherent and consistent text from the "`Perturbed Generation`" and integrate them into your continuation. Ensure a seamless alignment with the context established by the "`Prefix Text`". Provide only your "`Extended Text`".	`----"Prefix Text"`: {prompt} `----"Perturbed Generation"`: {perturbed prompt} `----"Extended Text":`

D Vocabulary Configuration

The vocabulary configuration for these original mechanisms are provided in Table 4. Note that CusText trims the GloVe's vocabulary (2,196,017) to align with a predefined vocabulary subset.

Table 4: Vocabulary configurations. $|\mathcal{V}|$ is the original vocabulary size; $|\mathcal{V}_{\text{sample}}|$ is the size of the mechanism's sampling space; K is the customized sampling space size in CusText. $d_{\max}$ is the maximum Euclidean distance between token embeddings in $|\mathcal{V}_{\text{sample}}|$.

Mechanisms	GloVe			Bert			GPT2		
	$\lvert\mathcal{V}\rvert$	$d_{\max}$	$\lvert\mathcal{V}_{\text{sample}}\rvert$	$\lvert\mathcal{V}\rvert$	$d_{\max}$	$\lvert\mathcal{V}_{\text{sample}}\rvert$	$\lvert\mathcal{V}\rvert$	$d_{\max}$	$\lvert\mathcal{V}_{\text{sample}}\rvert$
SanText	-	-	-	30,522	2.89	30,522	-	-	-
CusText	65,713	23.36	K	-	-	-	-	-	-
Cape(BERT)	-	-	-	30,522	2.89	30,522	-	-	-
Cape(GPT2)	-	-	-	-	-	-	50,257	7.58	50,257

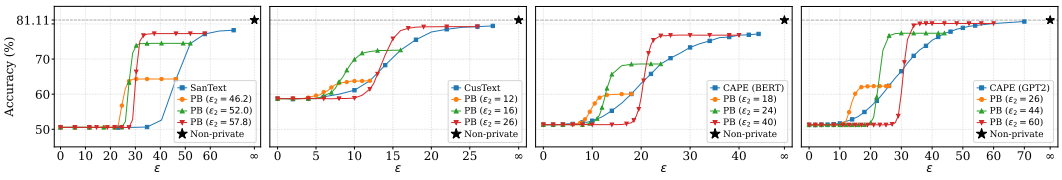
E Additional Experiment Results

E.1 Evaluation on utility

Figure 7 shows the utility measured by classification accuracy on the QNLI dataset. Again, it follows the S-shape of our sampling probability.

E.2 Evaluation on Privacy

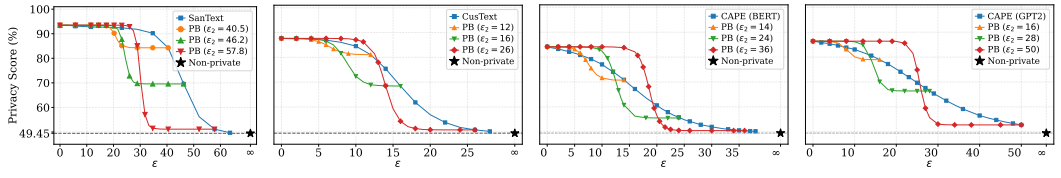
Results for MTI attack on the SST-2 dataset is presented in Figure 8, which is similar to Figure 3 under KNN attack presented in the main text. The difference is that privacy score of the unperturbed ($\varepsilon = \infty$) prompt is non-zero. This is because the BERT model for the attack cannot perfectly reconstruct the original prompt even without DP noise. The privacy-utility tradeoff on SST-2 measured by



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

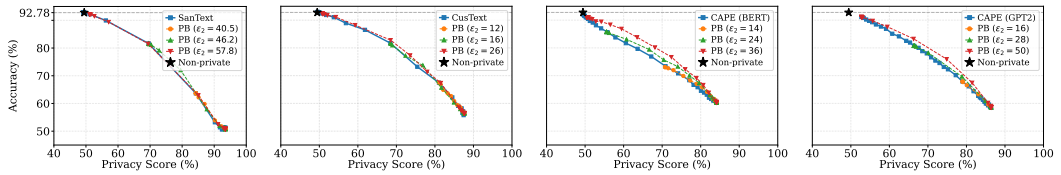
Figure 7: Utility on QNLI varing the privacy guarantee ε.

MTI attack score is presented in Figure 9. Again, we observe a similar tradeoff between empirical privacy score and utility measured by accuracy for all instantiations of PrivBlender.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

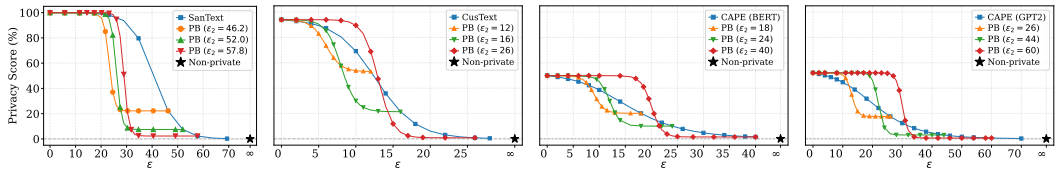
Figure 8: Privacy score under MTI Attack on SST-2



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

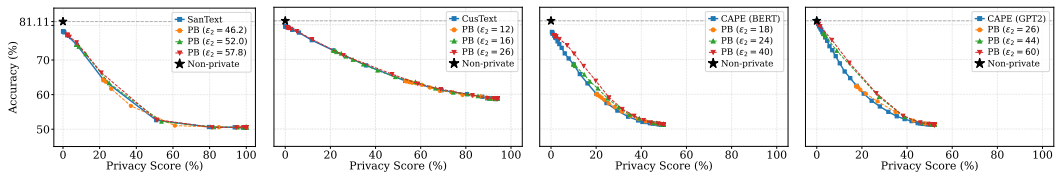
Figure 9: Privacy-utility tradeoff under MTI attack on SST-2.

Experiment results for the QNLI dataset are presented in Figure 10 to 13, where Figure 10 and 11 compares the empirical privacy score against nominal privacy budget ε and accuracy respectively, while Figure 12 and 13 shows corresponding results under the MTI attack.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

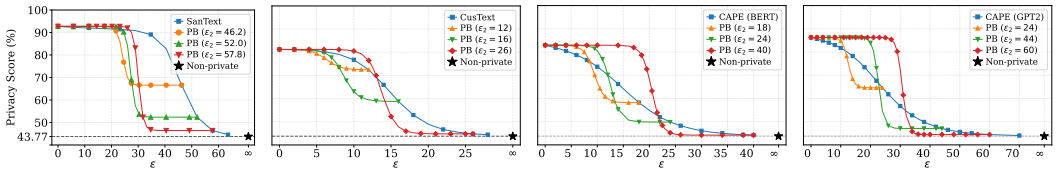
Figure 10: Privacy score under KNN attack on QNLI.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

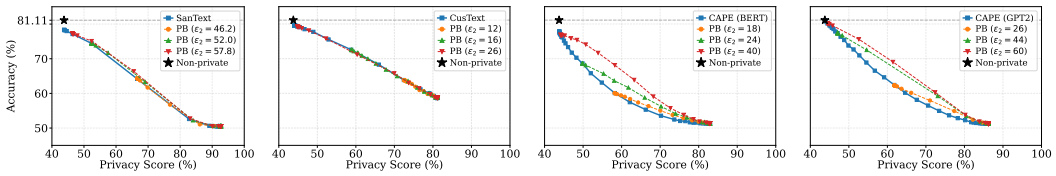
Figure 11: Privacy-utility tradeoff under KNN attack on QNLI.

For Privacy-utility tradeoff under MTI attack on QNLI with the underlying mechanism CAPE, as shown in Figure 13c and 13d, we observe a significant increase in utility for PrivBlender under the same empirical privacy score. This results from the convex shape of the utility-privacy curve of the baseline CAPE mechanism: since PrivBlender performs Bernoulli mixtures, it achieves an affine privacy-utility trade-off between the two candidate points.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

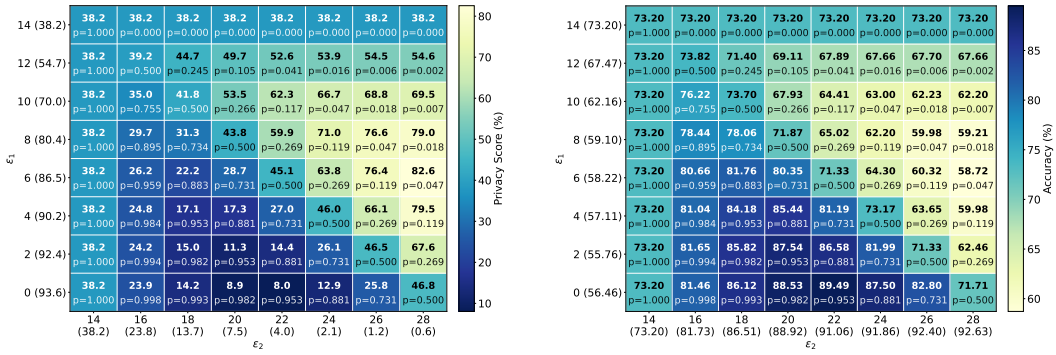
Figure 12: Privacy score under MTI Attack on QNLI.



(a) SanText (b) CusText (c) CAPE (BERT) (d) CAPE (GPT2)

Figure 13: Privacy-utility tradeoff under MTI attack on QNLI.

E.3 Offline Candidate Selection



(a) Privacy score under KNN attack varing ε_1 and ε_2 (b) Utility varing ε_1 and ε_2

Figure 14: Privacy score and utility on SST-2 with target $\varepsilon = 14$ using CusText as the baseline. Privacy (left) or utility (right) are labeled for each pair of $\varepsilon_{1,2}$ with the sampling probability p. The privacy or utility for the baseline mechanism is annotated near $\varepsilon_{1,2}$.