InferDPT: Privacy-preserving Inference for Black-box Large Language Models

Meng Tong, Kejiang Chen, Jie Zhang, Yuang Qi,
Weiming Zhang, Nenghai Yu, Tianwei Zhang, Zhikun Zhang

***Abstract*—Large language models (LLMs), represented by ChatGPT, have greatly simplified text generation tasks. However, they have also raised concerns about privacy risks such as data leakage and unauthorized information collection. Existing solutions for privacy-preserving inference face practical challenges related to computational time and communication costs. In this paper, we propose `InferDPT`, the first practical framework for privacy-preserving Infer<u>ence</u> of black-box LLMs, implementing Differential Privacy in Text generation. `InferDPT` comprises two key modules: the "perturbation module" utilizes the differentially private mechanism to generate a perturbed prompt, facilitating privacy-preserving inference with black-box LLMs; the "extraction module", inspired by knowledge distillation and phenomenon we observed, extracts coherent and consistent text from the perturbed generation result, ensuring successful text generation completion. To achieve a better balance between utility and privacy protection, we introduce RANTEXT, a novel differentially private mechanism integrated into the perturbation module of `InferDPT`, which introduces the concept of "RANdom adjacency list" for TEXT perturbation within the prompt. Experimental results across three datasets demonstrate that the text generation quality of `InferDPT` is comparable to that of non-private GPT-4, and RANTEXT surpasses existing state-of-the-art mechanisms, namely, SANTEXT+ and CUSTEXT+ in the trade-off between privacy and utility. Even with a privacy parameter ε value of 6.0, RANTEXT achieves an average privacy protection level of exceeding 0.90 against the embedding inversion attacks, which is 0.58$\times$ higher than that of SANTEXT+ and 3.35$\times$ higher than that of CUSTEXT+.**

***Index Terms*—Differential privacy, black box, inference, large language model.**

TABLE I
COMPARISONS OF DIFFERENT METHODS. A CHECK MARK (✓) INDICATES THAT METHODS MEET THE SCENARIO REQUIREMENTS.

Method	Text Generation	Black Box	Inference	Low Cost
CipherGPT [8]		✓	✓	
TextObfuscator [9]			✓	✓
DP-Forward [10]			✓	✓
SANTEXT+ [11]		✓		✓
CUSTEXT+ [12]		✓		✓
`InferDPT` + RANTEXT	✓	✓	✓	✓

I. INTRODUCTION

In recent years, the rapid advancement of *large language models* (LLMs) has garnered widespread attention from both the academic and industrial communities worldwide [1]. ChatGPT [2], a prominent example, has reached a remarkable milestone with 100 million weekly active users, as announced by OpenAI CEO Sam Altman on November 6, 2023, during the company's inaugural developer conference held in San Francisco [3]. The widespread popularity of ChatGPT has significantly facilitated people's daily work and lives. Users interact with ChatGPT via APIs or web interfaces to generate text for various applications, including but not limited to drafting articles, documenting daily work activities, and crafting advertisements for new products [4].

However, technology is a double-edged sword. While LLMs offer unparalleled convenience and utility in text generation, they may also raise potential privacy concerns. There are instances where the misuse of LLMs has led to serious privacy infringements. One such example involves Samsung employees leaking the company's confidential meeting records and sensitive data about unreleased products [5]. Furthermore, in a recent incident, GPT-3.5 unexpectedly disclosed an individual's selfies [6]. These incidents reignited concerns among the public regarding the potential privacy risks associated with uploading personal data to LLMs [7]. Therefore, it is crucial to address privacy concerns of uploading query contents, which is called *prompt*. We provide an example in Figure 1 to demonstrate privacy leakage in the *prompt* when a user interacts with LLMs.

Existing Solutions. A *prompt* in text generation tasks consists of a writing instruction and a document[1]. Previous studies [8]–[10] failed to protect the privacy within the document during the inference process in practical text generation tasks. As shown in Table I, CipherGPT [8] utilized homomorphic encryption techniques in transformer-architecture models to enable inference on encrypted data. While these techniques can be used theoretically for privacy-preserving text generation tasks, they have limitations in practical applications due to the significant computational time and communication costs. TextObfuscator [9] and DP-Forward [10] added noise during data transmission in *split learning*. However, they are mainly designed for classification tasks. Furthermore, they are unsuitable for black-box scenarios where the model owners, such as OpenAI [13], do not disclose details about

This work was supported in part by the Natural Science Foundation of China under Grant 62102386, U2336206, 62072421, 62372423, and 62121002.

M. Tong, K. Chen, Y. Qi, W. Zhang, and N. Yu are with CAS Key Laboratory of Electro-Magnetic Space Information, University of Science and Technology of China, Hefei 230026, China, and Anhui Province Key Laboratory of Digital Security. J. Zhang is with CFAR, A*STAR, T. Zhang is with Nanyang Technological University, and Z. Zhang is with Zhejiang University.

Corresponding author: Kejiang Chen (Email:chenkj@ustc.edu.cn) and Weiming Zhang (Email: zhangwm@ustc.edu.cn)

[1]The writing instruction provides directions on what the model should do; the document provides context that the model needs to generate a response.

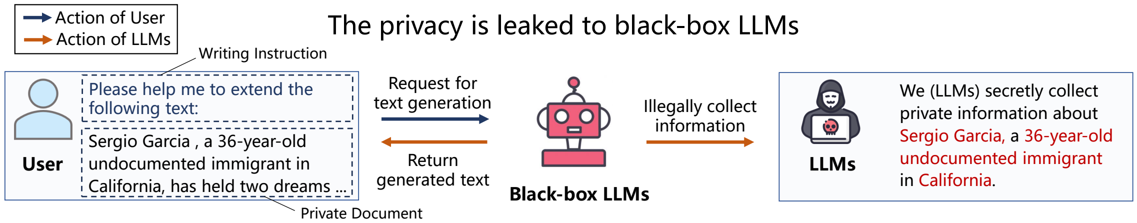

Fig. 1. The illustration of potential privacy leakage when a user employs black-box LLMs for text generation tasks.

the architectures of LLMs considering the intellectual property and commercial value of the models.

On the other hand, SANTEXT+ [11] and CUSTEXT+ [12] leveraged *local differential privacy* (LDP) techniques [14] to verbatim replace sensitive tokens in the text with semantically close tokens from a fixed token set, which is termed as *static adjacency list* in the LDP context. These methods are also designed for privacy-preserving classification tasks, which can tolerate considerable information distortion introduced by LDP noise. For the privacy-preserving text generation tasks [15] investigated in this paper, even a slight information distortion in the *prompt* can lead to incoherence and inconsistency in generated text, rendering SANTEXT+ and CUSTEXT+ not directly effective for such tasks. Additionally, the size of *static adjacency list* equals the entire vocabulary in SANTEXT+, which is excessively large and increases the probability that perturbed tokens are semantically irrelevant to raw ones. Moreover, our experimental results in Figure 8 demonstrate that CUSTEXT+ is vulnerable to embedding inversion attacks [16]: even in the extreme case where privacy parameter ε is set to 0.01, an adversary can still recover 40% of raw tokens in CUSTEXT+. The rationale behind this phenomenon is that each token has a small *static adjacency list* (which includes the raw token itself) in CUSTEXT+, thereby increasing the probability that the raw token will not be replaced.

Our Proposal. To protect the privacy of the entire document during the inference process with black-box LLMs and address the information bias caused by LDP, we introduce a framework, `InferDPT`, for text generation tasks. The general idea of `InferDPT` is derived from knowledge distillation [17] and our Observation supported by experimental results in Section IV-B: the generation of the perturbed *prompt* by LDP shares the same tokens in multiple parts of the generated text from the raw *prompt*. Furthermore, the number of the same tokens between them positively correlates with the privacy parameter ε. `InferDPT` comprises a perturbation module and an extraction module. In the perturbation module, `InferDPT` employs a differentially private mechanism, such as SANTEXT+ and CUSTEXT+, to perturb the raw document, obtaining a perturbed *prompt*. It uploads the perturbed *prompt* to remote LLMs. In the extraction module, `InferDPT` deploys a local model, lightweight and less capable than remote LLMs. This model extracts and reconstructs the perturbed generation result from the remote LLMs. Utilizing the perturbed generation result as reference and inferring texts according to the raw document, `InferDPT` not only safeguards the *prompt* privacy but also distills the capabilities of the remote large language model to improve text quality produced by the local model.

To achieve a better balance between utility and privacy protection, we develop RANTEXT. It is a novel differentially private mechanism integrated into the text perturbation of `InferDPT`. RANTEXT introduces the concept of *random adjacency list* for token-level perturbation. For each token, it employs the Laplace distribution [18] to dynamically determine the size of the *random adjacency list*, and then samples a new token from this list to replace raw tokens in the document. This approach enables RANTEXT to achieve a better trade-off between utility and privacy protection than existing methods do: (1) the *random adjacency list* in RANTEXT is typically smaller than SANTEXT+'s *static adjacency list*, which enhances the semantic utility of the perturbed text; (2) Compared with that in CUSTEXT+, the size of the *adjacency list* in RANTEXT is generally larger, making it more difficult for an adversary to reconstruct the raw tokens.

We conduct experiments on GPT-4 [13] for the evaluation of practical open-ended text generation tasks across three datasets. We found that existing attack strategies for differential privacy were not effective enough against RANTEXT. We propose an adaptive attack, the GPT inference attack, which leverages the capabilities of GPT-4 to reconstruct raw tokens.

Our Contributions. We summarize our main contributions:

- We propose `InferDPT`, the first practical framework for privacy-preserving inference of black-box large language models, implementing differential privacy in text generation.
- We develop RANTEXT, a novel exponential mechanism of local differential privacy integrated into document perturbation of `InferDPT`. It achieves a better balance between utility and privacy protection compared to existing baselines.
- We conduct experiments on three datasets tailored to practical open-ended text generation tasks in Section VI-B. Experimental results demonstrate that with ε set to 3.0 and a 3.89GB local model, `InferDPT` achieves generation quality comparable to GPT-4 in terms of three metrics.
- We evaluate four classes of privacy threats in Section VII. In particular, when we set the privacy parameter ε to 6.0 and select the top 10 candidates for embedding inversion attack, RANTEXT offers an average privacy protection level exceeding 0.90, which is 3.35× higher that of CUSTEXT+ and 0.58× higher than that of SANTEXT+.

II. Preliminaries

A. Large Language Models

Large language models (LLMs) are advanced artificial intelligence systems trained on extensive datasets. They are

TABLE II
NOTATIONS AND DEFINITIONS. WE INDICATE WHICH ELEMENTS ARE KNOWN TO THE ADVERSARY DURING THE INFERENCE.

Usr	User of LLMs
Adv	LLMs as an adversary
Ins	Instruction for text writing
Doc	Raw document of the user
Doc_p	Perturbed document of the user
Pro	Raw prompt of the user
Pro_p	Perturbed prompt of the user
Gen	Generation result of the raw prompt
Gen_p	Generation result of the perturbed prompt
V	Token vocabulary
C_r	Random adjacency list of token
C_e	Random adjacent embeddings of token
C_s	Static adjacency list of token
u	Scoring function of the exponential mechanism
M	Random mechanism of differential privacy
Infer	Inference function for text generation of LLMs

designed to understand, generate, and interpret human language, demonstrating incredible versatility for various language-related tasks. Generally, LLMs generate text Gen based on the prompt Pro uploaded by the users. They come in different types, including closed-source commercial services like ChatGPT [2] and Claude [19], as well as open-source models like Llama [20] and Vicuna [21]. In this paper, we focus on the closed-source LLMs and aim to address their privacy issues during the black-box inference in the open-ended text generation tasks.

Specifically, in the open-ended text generation task [22], the role of these black-box LLMs is to continue generating text Gen in accordance with the prompt Pro for higher text generation quality based on multi-dimensional metrics. In detail, given a prompt $Pro = Ins \parallel Doc$ consisting of Ins (fundamental writing instructions) and $Doc = \langle x_i \rangle_{i=1}^{L}$ (raw document composed of a sequence of L tokens x_i, belonging to token vocabulary V), the LLMs commit to providing inference function $\text{Infer}(\cdot): Pro \rightarrow Gen$ to generate text.

B. (Local) Differential Privacy& Exponential Mechanism

Differential privacy [14] is a privacy protection concept. As one of its most popular models, *ε-local differential privacy* (ε-LDP) allows data owners to locally perturb their data [23] using the randomized mechanism $M(\cdot)$ before uploading it to any untrusted aggregator.

Definition 1 (**ε-Local Differential Privacy [24]**). *In ε-LDP, given a privacy parameter $\varepsilon \geq 0$, a randomized mechanism M is ε-LDP compliant if it satisfies the following condition for any two inputs $x, x' \in X$ and any possible output $y \in Y$:*

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

Typically, a smaller value of ε provides higher privacy protection at the cost of reduced data utility. Moreover, a critical definition here is the input set X. In previous NLP research [11], [12], most researchers have posited that any pair of tokens in the vocabulary share the same input set X and output set Y. We observe that such a definition leads to a challenge in the trade-off between utility and privacy. In this paper, we use *random adjacency list* to redefine the input set of ε-LDP in Section V-B.

Definition 2 (**Exponential Mechanism [25]**). *For a given scoring function $u : X \times Y \rightarrow \mathbb{R}$, a randomized mechanism $M(\cdot)$ is ε-LDP compliant if it satisfies the following condition for any input $x \in X$ and any possible output $y \in Y$:*

$$\Pr[y|x] \propto \exp\left(\frac{\varepsilon \cdot u(x, y)}{2\Delta u}\right), \tag{2}$$

where the sensitivity Δu is defined as:

$$\Delta u = \max_{x, x' \in X, y \in Y} |u(x, y) - u(x', y)|. \tag{3}$$

The scoring function u is various in different scenarios. Typically, we can adjust the upper bound of u to set Δu to a specific real number, where Δu represents the sensitivity of the scoring function u. Similarly, the smaller the value of ε, the higher the security of privacy protection capability, but the lower the utility of the data. When a smaller ε is chosen, the scoring function $u(x, y)$ no longer plays a decisive role in the output probability of any perturbation result.

III. PROBLEM STATEMENT

A. Threat Model

We consider the scenario where the LLM platform, such as ChatGPT, is an honest but curious adversary, referred to as Adv. A user, denoted as Usr, intends to upload a prompt and invoke the inference service $\text{Infer}(\cdot): Pro \rightarrow Gen$ of Adv to complete the text generation tasks, which are poorly executed by open-source models. Here, Gen denotes the text generated by Adv. The uploaded prompt, $Pro = Ins \parallel Doc$ represents the raw prompt of Usr consisting of Ins (fundamental writing instructions) and $Doc = \langle x_i \rangle_{i=1}^{L}$ (raw document composed of a sequence of L tokens x_i, belonging to token vocabulary V).

Following previous works [9], [16], the privacy information probably pertains to each token. To protect each piece of the token in the raw document Doc, Usr employs differential privacy [14] to Doc, resulting in a perturbed document Doc_p. Consequently, Usr uploads the perturbed prompt $Pro_p = Ins \parallel Doc_p$. Furthermore, Usr can deploy a less capable language model than LLMs. To preserve the model's commercial value, Adv does not reveal the internal architecture or parameters of the LLMs, but only exposes its token vocabulary V to Usr for the purpose of billing verification during the inference process.

The goal of Adv is to reconstruct every piece of the token in the raw document Doc from Doc_p. Adv is expected to launch attacks using vulnerabilities in LDP, aiming to recover each token in the document Doc, based on the perturbed version Doc_p. Additionally, we assume that Adv is fully informed about the details of the differential privacy algorithm.

Table II summarizes notations frequently used in this paper.

B. Existing Solutions and Limitations

Existing solutions, such as SANTEXT+ [11] and CUSTEXT+ [12], focus on privacy-preserving model training in classification tasks:

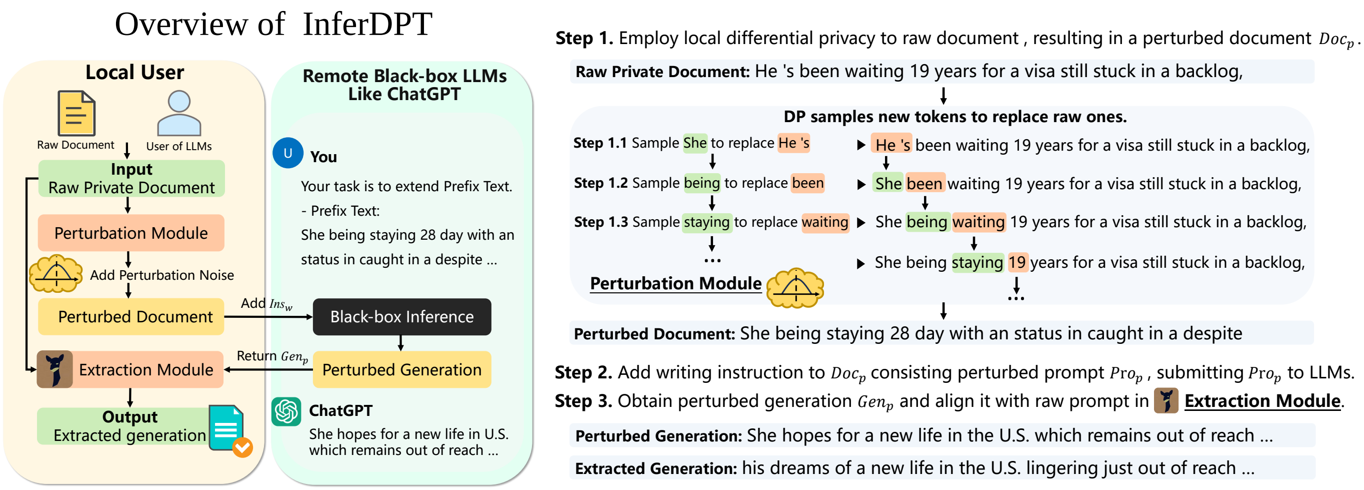

Fig. 2. The overview of `InferDPT`. It consists of (1) a perturbation module that samples new tokens to replace the raw ones in *Doc* via LDP and (2) an extraction module that locally aligns the perturbed generation with the raw document.

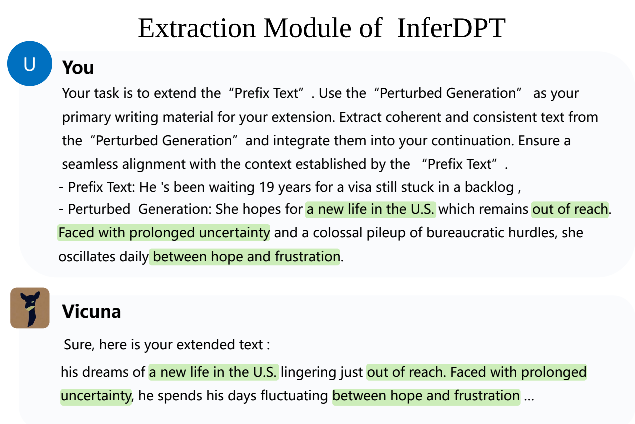

Fig. 3. The extraction module employs a smaller language model locally to extract text from the perturbed generation. It then reconstructs this text into an output that aligns with the raw document. We mark the text in green to indicate that it is identical in both the perturbed and extracted generations.

- SANTEXT+ implements local differential privacy (LDP) [26] during the training in classification tasks. It substitutes the raw tokens with newly sampled ones from a *static adjacency list*. However, this *static adjacency list* in SANTEXT+ equals the entire token vocabulary and is excessively large. Consequently, there exists a high probability that the perturbed token may be semantically irrelevant to the raw one, leading to diminished utility of the perturbed text.
- CUSTEXT+ perturbs each token, excluding stopwords [27], during training in classification tasks. Compared to SANTEXT+, it reduces the size of *static adjacency list* to a small number (default 20) for better utility of LDP. However, this small *static adjacency list* increases the probability that raw tokens will not be replaced, resulting in privacy leakage.

To protect the privacy of documents during inference in text generation tasks and address the information distortion introduced by LDP noise, we introduce a framework, `InferDPT`, (Section IV). We also propose an exponential mechanism, RANTEXT (see Section V), which offers a better trade-off between utility and privacy protection compared to existing SANTEXT+ and CUSTEXT+.

IV. The InferDPT Framework

A. Overview

We introduce `InferDPT`, a framework designed for privacy-preserving LLMs inference in text generation tasks. As shown in Figure 2, `InferDPT` is consisting of two modules:

- **Perturbation Module: protecting privacy**. It generates a perturbed document by replacing each token in *Doc* with one close to embedding distance and sampled by LDP.
- **Extraction Module: maintaining utility**. It extracts coherent and consistent text from perturbed generation and reconstructs them into an output aligned with the raw prompt by a local language model, less capable than black-box LLMs.

The design of `InferDPT` faces two main challenges in black-box inference. (1) **Providing strong privacy protection for the raw document** *Doc*. To solve this privacy challenge, the perturbation module of `InferDPT` utilizes a differentially private mechanism to sequentially replace each token in the raw document *Doc* with alternatives close in embedding distance. (2) **Maintaining the utility of the text under semantic perturbation**. This is more tough than the first one. To solve this challenge, we conducted abundant experiments about the generation of the perturbed document using LDP on LLMs. Specifically, we perturb each token in the document with a newly sampled one close in the embedding distance by LDP, resulting in a perturbed document. We discovered that the generation of this perturbed document includes numerous tokens found in the generation of the raw document. For example, we collect the tokens appearing in the generation of the raw document *Doc* in Figure 2, termed a set $\{hopes, new, dreams, \cdots\}$. We find that the generation of the perturbed document Doc_p (depicted in Figure 2) contains tokens within the set $\{hopes, new, dreams, \cdots\}$. Moreover, this overlap increases gradually as the perturbation decreases.

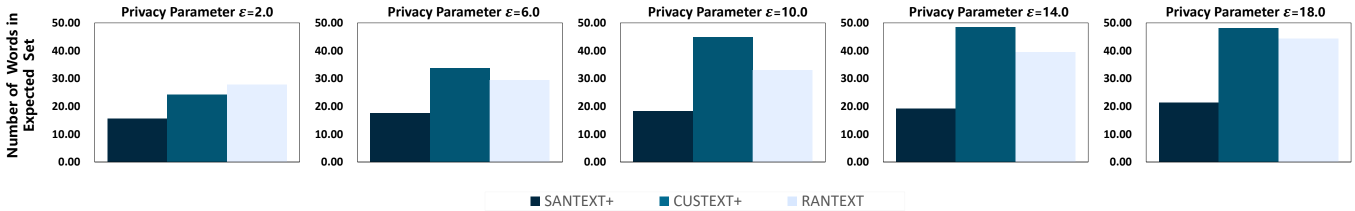

Fig. 4. The number of tokens from the non-private and private generation of GPT4 using three mechanisms that belong to the *Expected set*.

TABLE III
PROPORTION OF SYNONYMS BETWEEN THE RAW TOKENS AND THEIR PERTURBED VERSIONS [27].

Method	Synonym Proportion↓			
	$\varepsilon = 2.0$	$\varepsilon = 6.0$	$\varepsilon = 10.0$	$\varepsilon = 14.0$
SANTEXT+	0.371	0.373	0.374	**0.375**
CUSTEXT+	0.441	0.697	0.907	0.985
RANTEXT	**0.013**	**0.049**	**0.147**	0.378

TABLE IV
EUCLIDEAN DISTANCE BETWEEN THE EMBEDDINGS OF TOKENS AND THEIR PERTURBED VERSIONS.

Method	Euclidean Distance↑			
	$\varepsilon = 2.0$	$\varepsilon = 6.0$	$\varepsilon = 10.0$	$\varepsilon = 14.0$
SANTEXT+	3.081	2.775	2.756	2.750
CUSTEXT+	2.862	1.732	0.553	0.118
RANTEXT	**4.317**	**4.133**	**3.667**	**2.807**

To formally describe this phenomenon, we propose the following Observation about the perturbed output by LDP.

B. Key Observations

Observation. Let V be a token vocabulary. We define $d(\cdot)$ as a function that quantifies the semantic similarity between two tokens, where smaller output values indicate greater similarity. Let $M(\cdot)$ denote a randomized function of local differential privacy that satisfies:

$$d(x, y) \geq d(x, z) \Rightarrow \Pr[M(x) = y] \leq \Pr[M(x) = z], \quad (4)$$

where tokens $x, y, z \in V$.

Let tokens $x_i, y_i \in V$. $Doc_p = \langle y_i \rangle_{i=1}^{L}$ denotes the perturbed document of the raw document $Doc = \langle x_i \rangle_{i=1}^{L}$ by $y_i = M(x_i)$. The expression $\text{Infer}(Ins \parallel Doc) = \langle h_i^{(j)} \rangle_{i=1}^{K}$ represents the generation result of the j-th inference on Doc, consisting of tokens $h_i^{(j)} \in V$. Given Doc_p, the perturbed generation $Gen_p = \langle g_i \rangle_{i=1}^{K} = \text{Infer}(Ins \parallel Doc_p)$ consists of tokens $g_i \in V$ and satisfies the following condtion:

$$\textit{Expected set} = \bigcup_{j=1}^{N} \{h_i^{(j)} \,|\, h_i^{(j)} \in \text{Infer}(Ins \parallel Doc)\}, \quad (5)$$

$$\textit{Intersection} = \{g_i | g_i \in \textit{Expected set} \text{ and } g_i \notin \textit{stopwords}^2\}, \quad (6)$$

$$\text{Corr}(\text{Count}(\textit{Intersection}), \varepsilon) > 0, \quad (7)$$

where N is a positive integer; the function $\text{Count}(\cdot)$ *counts the size of a set; the function* $\text{Corr}(\cdot)$ *measures correlation coefficient between two variables, with values ranging from -1 (negative correlation) to 1 (positive correlation).*

Implication. This Observation states that if the *Expected set* is constructed from tokens in the results of N iterations of raw prompt, then the presence of tokens from the perturbed generation within the *Expected set* will positively correlate with ε. This implies that smaller perturbations to Doc lead to higher consistency between the perturbed generation and the raw generation. To verify this Observation, we carried out the following experiments with GPT-4 [13].

[2]Stopwords [27] are common words usually ignored in text analysis due to their limited informational values.

Empirical Validation. We got the $Expected\ set$ by collecting 100 tokens from the output of the raw prompt with GPT-4 generated 100 times on the CNN/Daily Mail dataset [28]. The raw prompt consists of a fundamental writing instruction and a raw document of 50 tokens shown in Figure 2. We utilized SANTEXT+ [11], CUSTEXT+ [12], and RANTEXT introduced in Section V to generate perturbed outputs of 100 tokens from GPT-4 under various values of ε. We counted the number of tokens from the perturbed and non-private generation of GPT-4 that belong to the $Expected\ set$.

Figure 4 shows the experimental results. We can see that with the increase of ε and reduction of perturbation, the number of tokens in the $Expected\ set$ of the three mechanisms has increased. This validates Observation, confirming that the number of tokens from the $Expected\ set$ appearing in the perturbed generation positively correlates with ε.

Based on this Observation, `InferDPT` only needs an extraction module to extract and reconstruct perturbed output from the remote black-box LLMs, distilling its capabilities to accomplish text generation tasks. In the following subsections, we will delve into these two modules of `InferDPT`.

C. Perturbation Module

The perturbation module of `InferDPT` generates a perturbed document Doc_p from an input document Doc. Specifically, it replaces each token in Doc with new tokens sampled by the randomized mechanism $M(\cdot)$ of LDP from a *token set*. In previous works [11], [12], this *token set* is static (termed the *static adjacency list* C_s), whereas in our proposed RANTEXT, it is random (termed the *random adjacency list* C_r). Typically, C_s (or C_r) consists of tokens that are close to the embedding of the token $x_i \in V$ to be replaced. Given a document $Doc = \langle x_i \rangle_{i=1}^{L}$ composed of L tokens $x_i \in V$, the perturbation module replaces each x_i with a random output $y_i = M(x_i, C_s(x_i))$ or $y_i = M(x_i, C_r(x_i))$, resulting in a perturbed document $Doc_p = \langle y_i \rangle_{i=1}^{L}$ (as shown in Figure 2). The detailed process of the perturbation module is outlined in

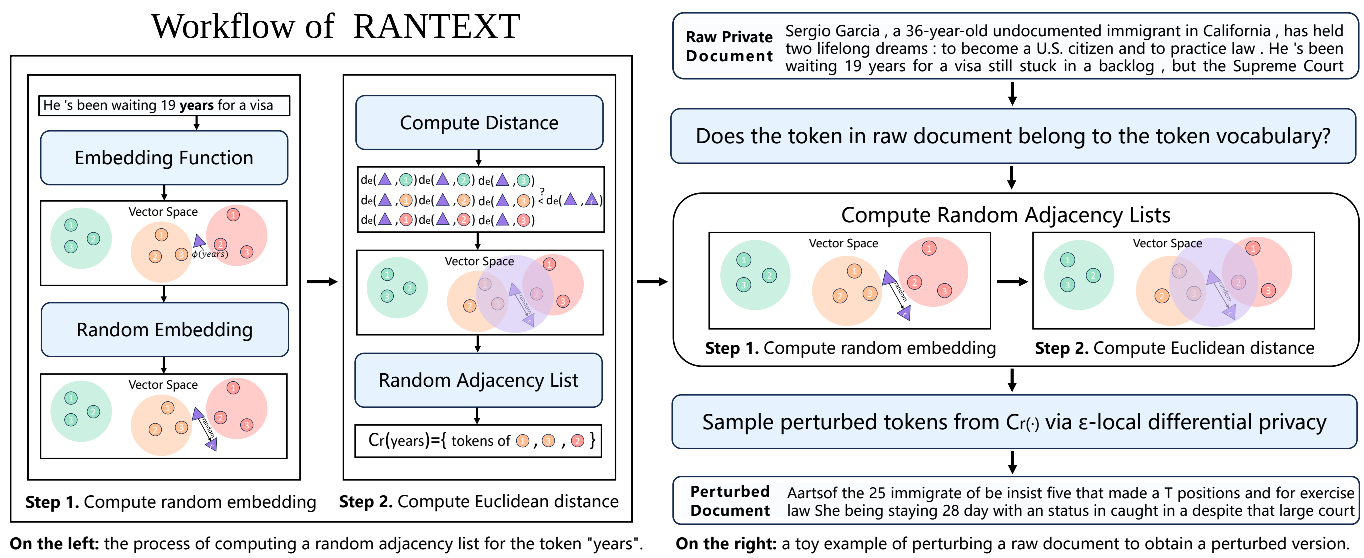

Fig. 5. The workflow of RANTEXT. It comprises two steps: (1) computing random adjacency lists and (2) sampling perturbed tokens via ε-LDP.

Algorithm 1 Perturbation Module

Input: Document $Doc = \langle x_i \rangle_{i=1}^{L}$, random mechanism $M(\cdot)$, static adjacency list $C_s(\cdot)$, *random adjacency list* $C_r(\cdot)$;
Output: Perturbed document Doc_p;
Initialize $Doc_p \leftarrow \emptyset$;
for $i = 1$ *to* L **do**
  Compute $C_s(x_i)$ or $C_r(x_i)$;
  Sample $y_i \sim M(x_i, C_s(x_i))$ or $y_i \sim M(x_i, C_r(x_i))$;
  Append y_i to Doc_p;
end for
Output Doc_p;

Algorithm 1. In the implementations, `InferDPT` adopts three mechanisms of LDP: SANTEXT+ [11], CUSTEXT+ [12], and RANTEXT (detailed in the following Section V).

While LDP perturbs sensitive text, it is important to note that an excessively large ε in LDP increases the risk of privacy leakage from the perturbed text. This is because LDP perturbs raw tokens to more semantically close tokens as ε increases. We experimentally demonstrate this risk by calculating the synonymous token proportion and the embedding distance (between the raw tokens and their perturbed tokens) with various ε values. As shown in Table III and Table IV, as ε increases, the synonym proportion grows and the Euclidean distance decreases, indicating greater semantic similarity between the raw tokens and the perturbed tokens.

After perturbation, Usr uploads a perturbed prompt $Pro_p = Ins \parallel Doc_p$ (consisting of a writing instruction Ins and a perturbed document Doc_p) to remote LLMs. The LLMs then return perturbed generation $Gen_p = \text{Infer}(Pro_p)$ to Usr.

D. Extraction Module

As previously mentioned, the perturbation module disturbs each token and key information in the raw document Doc, making it difficult for an adversary to reconstruct the raw tokens from Doc_p or Gen_p. However, this perturbation also leads to inconsistency and partial incoherence of semantics between Gen_p and Doc, as illustrated in Figure 2.

To obtain the aligned generation of the raw document Doc, the extraction module of `InferDPT` deploys a local language model that is considered trustworthy and does not pose any privacy leakage issues. This local model is smaller and less powerful than remote black-box LLMs, facilitating easier implementation under limited resources. As shown in Figure 3, Usr inputs the raw document Doc and the perturbed generation Gen_p into this local model. This model is tasked with extracting coherent and consistent text from Gen_p and integrating it into the continuation of Doc, ensuring an aligned output. Although the local model can generate aligned content independently, the generation quality is not satisfactory due to its limited capabilities. However, with the perturbed generation Gen_p, the local model distills the capacity of the remote black-box LLMs. The details of the prompt utilized in the extraction module can be found in Appendix A.

Based on the above description, we have a panoramic view of `InferDPT`. It is noted that the perturbation module can adopt existing differentially private mechanisms such as SANTEXT+ [11] and CUSTEXT+ [12]. However, these two have drawbacks either in terms of utility or security, as analyzed in Section III-B. To address these problems, we introduce RANTEXT in the following section.

V. The RANTEXT Mechanism

A. Overview

We design RANTEXT to address the utility and vulnerability problems of previous differentially private mechanisms [11], [12]. As shown in Figure 5, RANTEXT comprises two steps:

- **Compute Random Adjacency Lists.** This step computes a *random adjacency list* for each raw token via two operations: computing random embedding and Euclidean distance. Any tokens in *random adjacency list* share the same input set.
- **Sample Perturbed Tokens via ε-LDP.** This step samples a perturbed token for each raw token and replaces the raw token in the document from its *random adjacency list* via ε-LDP, obtaining the perturbed document.

As mentioned in Section III-A, LLMs expose their token vocabulary V for billing verification of inference service. Utilizing the token vocabulary V and Byte Pair Encode (BPE) algorithm [29], users can obtain the $tokenizer(\cdot)$ of LLMs.

Given a raw document Doc, RANTEXT first uses the $tokenizer(\cdot)$ algorithm of LLMs to turn the text of Doc into tokens $\langle x_i \rangle_{i=1}^{L}$, where $x_i \in V$:

$$Tokenset = \langle x_i \rangle_{i=1}^{L} = tokenizer(Doc). \tag{8}$$

To preserve the privacy of Doc, RANTEXT discards the tokens of Doc that do not belong to V and employs an exponential mechanism to subsequently replace each remaining token with one close in embedding distance from its exclusive *random adjacency list*:

$$r_i = M(x_i, C_r(x_i)), \tag{9}$$

$$Tokenset_p = \langle r_i \rangle_{i=1}^{l} = \langle M(x_i, C_r(x_i)) \rangle_{i=1}^{l}, \tag{10}$$

where token $r_i \in V$, $Tokenset_p$ represents the perturbed token set, and $C_r(x_i)$ represents the *random adjacency list* of x_i. RANTEXT concatenates the tokens in a perturbed token set $Tokenset_p$ to obtain a perturbed document Doc_p, thereby providing privacy protection.

B. Compute Random Adjacency Lists

To formally define the *random adjacency list*, we first give a definition of *random adjacent embeddings*:

Definition 3 (**Random Adjacent Embeddings**). *Given token $t \in V$, its random adjacent embeddings are defined as follows:*

$$C_e(t) = \{eb | d_e(eb, \phi(t)) < d_e(\hat{\phi}(t), \phi(t)), eb \in \mathbb{R}^N\}, \tag{11}$$

where $eb \in \mathbb{R}^N$ represents any N-dimensional vector within the real number domain. The function $d_e(\cdot)$ is utilized to compute the distance between two vectors and is defined as $d_e(a, b) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$. The function $\phi : V \to \mathbb{R}^N$ maps any given token to a vector in the N-dimensional real number vector space. The function $\hat{\phi}(t) = \phi(t) + Y$, where the random vector Y satisfies the probability density:

$$Y \sim f(x) = \frac{Z}{2\Delta\phi} \cdot \exp\left(-\frac{Z \cdot |x|}{\Delta\phi}\right), \tag{12}$$

$$Z = \begin{cases} \varepsilon & \text{if } \varepsilon < 2, \\ a \log(b \cdot \varepsilon + c) + d & \text{otherwise}, \end{cases} \tag{13}$$

where $\Delta\phi$ is the sensitivity of function $\phi(\cdot)$, $a \approx 0.0165$, $b \approx 19.0648$, $c \approx -38.1294$, $d \approx 9.3111$.

Given a token $t \in V$ to compute its random adjacent embeddings, we need to complete the two-step computation:
Step 1. Compute the random embedding. We construct the random vector Y utilizing the Laplace distribution [18]. We add Y independently to each dimension of $\phi(t)$, obtaining the random embedding $\hat{\phi}(t) = \phi(t) + Y$ of raw private token t.
Step 2. Compute the Euclidean distance. We compute the Euclidean distance between $\phi(t)$ and $\hat{\phi}(t)$, referred to $d_e(\hat{\phi}(t), \phi(t))$. The random adjacent embeddings consist of those embeddings whose Euclidean distance to $\phi(t)$ is shorter than $d_e(\hat{\phi}(t), \phi(t))$.

Algorithm 2 RANTEXT Mechanism

Input: Token set $Tokenset = \langle x_i \rangle_{i=1}^{L}$, token vocabulary V, privacy parameter ε, embedding function $\phi(\cdot)$, distance function $d_e(\cdot)$, random vector Y;
Output: Perturbed document Doc_p;

```
Initialize Tokenset_p ← ∅;
for i = 1 to L do
    if x_i ∉ V then
        Discard the token x_i ;
        Continue;
    end if
    Sample a random vector Y;
    Compute embedding eb_t ← φ(x_i);
    Compute random embedding eb_n ← eb_t + Y;
    Compute Euclidean distance d_threshold ← d_e(eb_n, eb_t);
    C_e(x_i) = {eb | d_e(eb, eb_t) < d_threshold , eb ∈ ℝ^N};
    C_r(x_i) = {x_i' | φ(x_i') ∈ C_e(x_i), x_i' ∈ V};
    for each x_i' ∈ C_r(x_i) do
        d_{x_i'} ← d_e(φ(x_i), φ(x_i'));
        Scoring function u(x_i, x_i') ← 1 − d_{x_i'}/d_threshold;
        p_total ← p_total + exp(ε/2 · u(x_i, x_i'));
    end for
    for each x_i'' ∈ C_r(x_i) do
        p(x_i''|x_i) ← exp(ε/2 · u(x_i, x_i'')) / p_total;
    end for
    Sample from random adjacency list r_i ~ p(x_i''|x_i);
    Append new token r_i to perturbed token set Tokenset_p;
end for
Concatenate Tokenset_p = ⟨r_i⟩_{i=1}^{L} obtaining Doc_p
Output perturbed document Doc_p;
```

We use Y to dynamically determine the size of the *random adjacency list*. The detailed construction process of the random vector Y can be found in Appendix B.

With the definition of random adjacent embeddings, we give the definition of the *random adjacency list*:

Definition 4 (**Random Adjacency List**). *Given a token $t \in V$, its random adjacency list is defined as follows:*

$$C_r(t) = \{t' | \phi(t') \in C_e(t), t' \in V\}. \tag{14}$$

Given a token $t \in V$, its *random adjacency list* is composed of any token t' in the token vocabulary V, whose embedding $\phi(t')$ has a Euclidean distance to $\phi(t)$ shorter than the Euclidean distance between t's random embedding and t's embedding $\phi(t)$.

The design of the *random adjacency list* in RANTEXT obeys the following theorem:

Theorem 1. *Given a token $t \in V$ and any token $t' \in V$, there exists a random adjacency list $C_r(t)$ of RANTEXT satisfying $t' \in C_r(t)$.*

Theorem 1 is proven in Appendix C. It demonstrates that a token t can be substituted with any token $t' \in V$ in RANTEXT, thereby increasing the difficulty for adversaries to reconstruct the raw tokens. Moreover, the *random adjacency list* addresses the utility problem of the perturbed text in

TABLE V
PERFORMANCE COMPARISON ON OPEN-ENDED TEXT GENERATION TASKS ACROSS DIFFERENT METHODS, DATASETS, AND PRIVACY PARAMETERS ($\varepsilon = 1, 2, 3$), EVALUATED BASED ON DIVERSITY, MAUVE, AND COHERENCE.

Dataset	Method	diversity↑			MAUVE↑			coherence↑		
		$\varepsilon = 1.0$	$\varepsilon = 2.0$	$\varepsilon = 3.0$	$\varepsilon = 1.0$	$\varepsilon = 2.0$	$\varepsilon = 3.0$	$\varepsilon = 1.0$	$\varepsilon = 2.0$	$\varepsilon = 3.0$
CNN/Daily Mail	GPT-4		0.983			0.671			0.632	
	Vicuna-7b-4bit (3.89GB)		0.943			0.197			0.627	
	InferDPT + SANTEXT+	0.966	0.967	0.966	0.351	0.374	0.407	0.590	0.632	0.642
	InferDPT + CUSTEXT+	0.966	0.967	0.965	0.540	**0.571**	0.581	**0.726**	0.733	**0.752**
	InferDPT + RANTEXT	**0.970**	**0.970**	**0.971**	**0.542**	0.563	**0.587**	0.723	**0.735**	0.736
Wikitext-103-v1	GPT-4		0.987			0.453			0.672	
	Vicuna-7b-4bit (3.89GB)		0.916			0.158			0.663	
	InferDPT + SANTEXT+	0.958	0.958	0.959	0.213	0.220	0.255	0.650	0.658	0.678
	InferDPT + CUSTEXT+	0.960	0.961	0.959	**0.301**	**0.315**	**0.321**	0.727	0.736	0.741
	InferDPT + RANTEXT	**0.961**	**0.962**	**0.961**	0.245	0.254	0.274	**0.729**	**0.744**	**0.745**
ArXiv Dataset	GPT-4		0.935			0.736			0.726	
	Vicuna-7b-4bit (3.89GB)		0.873			0.366			0.703	
	InferDPT + SANTEXT+	0.945	0.946	0.946	0.196	0.207	0.230	0.651	0.670	0.690
	InferDPT + CUSTEXT+	0.946	0.945	0.944	**0.410**	**0.443**	**0.455**	0.748	**0.767**	**0.784**
	InferDPT + RANTEXT	**0.947**	**0.948**	**0.947**	0.359	0.375	0.395	**0.752**	0.761	0.762

TABLE VI
PERFORMANCE COMPARISON OF THE TIME COST PER INFERENCE ON 100 TOKENS IN INFERDPT.

Method	Time Cost (seconds)		
	SANTEXT+	CUSTEXT+	RANTEXT
Perturbation Module	0.0015 ± 0.0001	0.0005 ± 0.0001	0.0543 ± 0.0023
Black-box Inference	-	2.8324 ± 0.2111	-
Extraction Module	-	3.5673 ± 0.2781	-

SANTEXT+. Although the theoretically maximum size of the *random adjacency list* is equivalent to the size of V, it is typically smaller than that in terms of probability. This reduces the likelihood that the perturbed token is semantically irrelevant.

Furthermore, experimental results in the following Section VI-B demonstrate that the *random adjacency list* in RANTEXT is generally larger than the *static adjacency list* in CUSTEXT+, which solves the vulnerability of CUSTEXT+ [12] to the embedding inversion attack.

C. Sampling Perturbed Tokens via ε-LDP

In SANTEXT+ [11], a proportion of tokens is not perturbed by LDP. To solve the privacy leakage issue in the raw text, RANTEXT perturbs every piece of the token in $Tokenset = \langle x_i \rangle_{i=1}^{L}$. To perturb token x_i, RANTEXT employs the exponential mechanism [25], which satisfies ε-LDP, to select a new token from $C_r(x_i)$ to replace the original one. For any special token $t_s \notin V$, RANTEXT discards it, to ensure there is no special token leakage in Doc_p.

To guarantee the utility of the perturbed document, the random mechanism $M(\cdot)$ of the exponential mechanism in RANTEXT is required to satisfy:

$$d(x, y) \geq d(x, z) \Rightarrow Pr[M(x) = y] \leq Pr[M(x) = z], \quad (15)$$

where $x \in V$, and y and z belong to the random adjacency list of x. $d(\cdot)$ measures the semantic similarity between two inputs, with a smaller output indicating greater similarity.

To fulfill that, the scoring function $u(\cdot)$ of the random mechanism $M(\cdot)$ in RANTEXT is described as follows:

Given a token t, RANTEXT considers that any two tokens in $C_r(t)$ share the same input set and output set during the perturbation of token t. Given any two tokens $x, y \in C_r(t)$, the scoring function is

$$u(x, y) = 1 - \frac{|d_e(\phi(x), \phi(t)) - d_e(\phi(y), \phi(t))|}{d_e(\phi(t), \hat{\phi}(t))}. \quad (16)$$

With Equation 11 and Equation 14, it holds that:

$$|d_e(\phi(x), \phi(t)) - d_e(\phi(y), \phi(t))| < d_e(\phi(t), \hat{\phi}(t)), \quad (17)$$

$$0 \ \leq \ \frac{|d_e(\phi(x), \phi(t)) - d_e(\phi(y), \phi(t))|}{d_e(\phi(t), \hat{\phi}(t))} < \ 1 \ . \quad (18)$$

With Equation 16 and Equation 17, it can be deduced that:

$$0 \ < u(x, y) \leq \ 1 \ , \quad (19)$$

$$\Delta u = 1 \ . \quad (20)$$

Given a privacy parameter $\varepsilon \geq 0$, the probability of obtaining an output of the perturbed token $y \in C_r(t)$ for any input token $x \in C_r(t)$ is as follows:

$$Pr[y|x] = \frac{\exp\left(\frac{\varepsilon \cdot u(x,y)}{2\Delta u}\right)}{\sum_{y' \in C_r(t)} \exp\left(\frac{\varepsilon \cdot u(x,y')}{2\Delta u}\right)} \quad (21)$$

$$= \frac{\exp\left(\frac{\varepsilon}{2} \cdot \left(1 - \frac{|d_e(\phi(x),\phi(t)) - d_e(\phi(y),\phi(t))|}{d_e(\phi(t),\hat{\phi}(t))}\right)\right)}{\sum_{y' \in C_r(t)} \exp\left(\frac{\varepsilon}{2} \cdot \left(1 - \frac{|d_e(\phi(x),\phi(t)) - d_e(\phi(y'),\phi(t))|}{d_e(\phi(t),\hat{\phi}(t))}\right)\right)}. \quad (22)$$

Specifically for the input token $t \in C_r(t)$ and output token $y \in C_r(t)$, it can be deduced that:

$$u(t, y) = 1 - \frac{d_e(\phi(y), \phi(t))}{d_e(\phi(t), \hat{\phi}(t))}, \quad (23)$$

$$Pr[y|t] = \frac{\exp\left(\frac{\varepsilon}{2} \cdot \left(1 - \frac{d_e(\phi(t),\phi(y))}{d_e(\phi(t),\hat{\phi}(t))}\right)\right)}{\sum_{y' \in C_r(t)} \exp\left(\frac{\varepsilon}{2} \cdot \left(1 - \frac{d_e(\phi(t),\phi(y'))}{d_e(\phi(t),\hat{\phi}(t))}\right)\right)}. \quad (24)$$

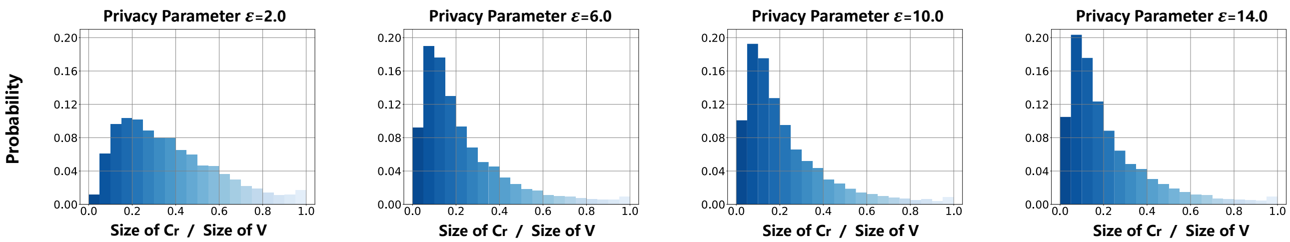

Fig. 6. Probability distribution of the size of *random adjacency list* under various ε.

TABLE VII
COMPARISON OF THE FINAL GENERATED TEXT QUALITY UNDER DIFFERENT LOCAL MODELS WITHIN THE EXTRACTION MODULE.

Dataset	Method	diversity↑			MAUVE↑			coherence↑		
		$\varepsilon=2.0$	$\varepsilon=6.0$	$\varepsilon=10.0$	$\varepsilon=2.0$	$\varepsilon=6.0$	$\varepsilon=10.0$	$\varepsilon=2.0$	$\varepsilon=6.0$	$\varepsilon=10.0$
CNN/Daily Mail	GPT-4		0.983			0.671			0.632	
	Llama2-7b-4bit (3.79GB)		0.896			0.258			0.485	
	Vicuna-7b-4bit (3.89GB)		0.943			0.197			0.627	
	SANTEXT$^+$(Llama2-7b-4bit)	0.964	0.963	0.962	0.282	0.374	0.406	0.226	0.327	0.342
	CUSTEXT$^+$(Llama2-7b-4bit)	0.963	0.962	0.963	**0.493**	0.519	0.548	**0.460**	**0.483**	0.514
	RANTEXT (Llama2-7b-4bit)	**0.968**	**0.969**	**0.967**	0.473	**0.526**	**0.566**	0.411	0.453	**0.525**
	SANTEXT$^+$(Vicuna-7b-4bit)	0.967	0.969	0.968	0.374	0.413	0.448	0.632	0.679	0.727
	CUSTEXT$^+$(Vicuna-7b-4bit)	0.966	0.967	0.968	**0.571**	**0.632**	**0.670**	0.733	0.749	**0.789**
	RANTEXT (Vicuna-7b-4bit)	**0.970**	**0.969**	**0.970**	0.563	0.586	0.635	**0.735**	**0.753**	0.773

TABLE VIII
COSINE SIMILARITY↑ BETWEEN THE FINAL GENERATION OF INFERDPT AND THE NON-PRIVATE GENERATION FROM GPT-4.

Dataset	Method	ε 1.0	2.0	3.0
CNN/Daily Mail	SANTEXT$^+$	0.489	0.499	0.519
	CUSTEXT$^+$	0.571	**0.579**	**0.585**
	RANTEXT	**0.574**	**0.579**	0.584
Wikitext-103-v1	SANTEXT$^+$	0.544	0.546	0.572
	CUSTEXT$^+$	0.597	**0.613**	**0.627**
	RANTEXT	**0.598**	0.609	0.617
ArXiv Dataset	SANTEXT$^+$	0.584	0.591	0.595
	CUSTEXT$^+$	**0.682**	**0.693**	**0.694**
	RANTEXT	0.655	0.658	0.663

The detailed process of RANTEXT is shown in Algorithm 2. Furthermore, the token sampling for each raw token in RANTEXT satisfies the definition of ε-LDP:

Theorem 2. *Given a privacy parameter $\varepsilon \geq 0$ and a random adjacency list $C_r(t)$ of token t, for any input tokens $x, x' \in C_r(t)$ and output token $y \in C_r(t)$, the randomized mechanism M of RANTEXT holds that:*

$$\frac{\Pr[M(x)=y]}{\Pr[M(x')=y]} \leq e^{\varepsilon}. \tag{25}$$

Theorem 2 demonstrates that given a $C_r(t)$ of token t, the token sampling for each raw token in RANTEXT satisfies ε-LDP. Theorem 2 is proven in Appendix C.

VI. EXPERIMENTS

A. Experiment Setup

Datasets. For the open-ended text generation tasks, we use two classic NLP datasets: CNN/Daily Mail [28] for news articles and Wikitext-103-v1 [30] for Wikipedia articles. For practical applications, we use ArXiv Dataset [31] for drafting scientific papers. These datasets cover a lot of events and individuals.

Baselines. `InferDPT` is the first practical framework for privacy-preserving inference that implements differential privacy in text generation tasks [32]. As there are no other frameworks of the same type, we did not compare `InferDPT` with any others. For the differentially private mechanisms of the perturbation module, we compared RANTEXT with existing state-of-the-art mechanisms, SANTEXT+ [11] and CUSTEXT+ [12] in the default settings of them.

Metrics. Following previous works of open-ended text generation [22], [32], we use the first 50 tokens of the articles referred to raw document Doc, which we must protect. We use the continuation writing of Doc referred to as Gen, which consists of 100 tokens. Tokens are counted by the tokenizer function of GPT-2 [33]. Aligning with [34], three metrics were employed to evaluate the quality of the generated text in the open-ended generation task:
1) *Diversity.* This metric suggests the text's diversity by computing the unique n-gram repetition rates as follows:

$$diversity = \sum_{n=2}^{4} \frac{|unique\ n\!-\!grams(Gen)|}{|total\ n\!-\!grams(Gen)|}.$$

A lower score indicates that the model is prone to repetition, while a higher score shows broader vocabulary usage.
2) *MAUVE* [35]. It is employed to assess the similarity between text generated by a language model and human-authored target continuation text. A higher score is desirable in this metric.
3) *Coherence.* Coherence computes the cosine similarity between embeddings of document Doc and continuation Gen:

$$COH(Doc, Gen) = \frac{\text{SimCSE}(Doc)\cdot\text{SimCSE}(Gen)}{\|\text{SimCSE}(Doc)\|\cdot\|\text{SimCSE}(Gen)\|},$$

where SimCSE(x) represents the pretrained model [36].

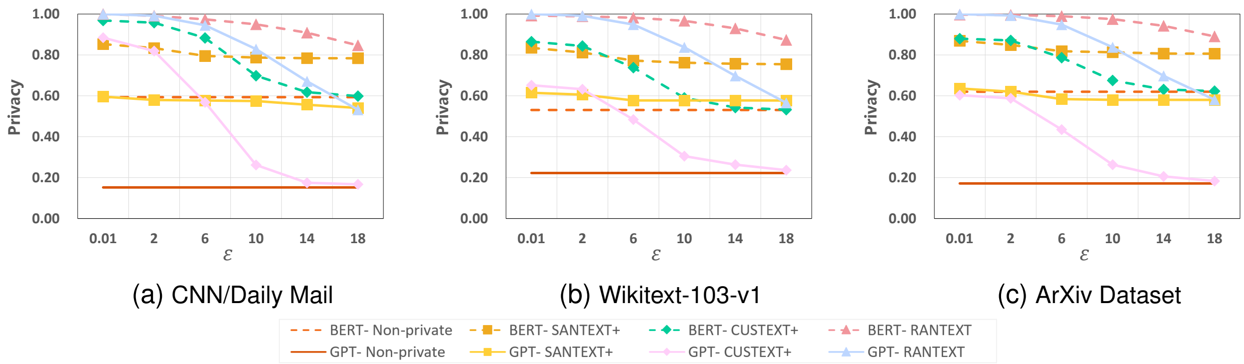

Fig. 7. Results of BERT inference attack and GPT inference attack on CNN/Daily Mail, Wikitext-103-v1, and ArXiv Dataset.

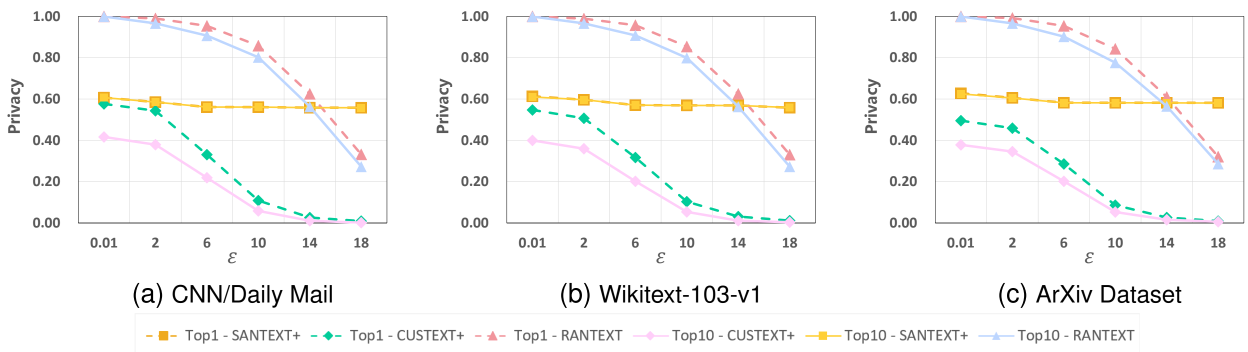

Fig. 8. Results of embedding inversion attack on CNN/Daily Mail, Wikitext-103-v1 and ArXiv Dataset.

Implementation. We run experiments on a cluster with NVIDIA RTX A6000 GPUs and Intel Xeon Gold 6130 2.10 GHz CPUs. We have conducted the implementation of all mechanisms using Python. For the black-box inference, we use GPT-4 [13] as our remote large language model with the temperature parameter set to 0.5. Correspondingly, GPT-4's token vocabulary is cl100k_base [37]. We use the initial 11,000 English tokens of cl100k_base as V. For the embedding function $\phi(\cdot)$, we choose text-embedding-ada-002 [38], which utilizes the same token vocabulary cl100k_base as employed during the training of GPT-4. We use Vicuna-7b-4bit [39] and Llama2-7b-4bit [20] as the local language model of the extraction module with the temperature set to 0.5.

B. Evaluation of Utility

We evaluated the quality of outputs generated by `InferDPT` with various differentially private mechanisms in the perturbation module, using the Vicuna-7b-4bit (3.89GB) in the extraction module on various datasets. Table V shows `InferDPT`'s generation quality compared to non-private GPT-4:

(1) Although the uploaded prompt is perturbed by differential privacy, the quality of text generated by `InferDPT` is comparable to that directly produced by non-private GPT-4 and better than the local model's output. It proves that `InferDPT` works effectively. (2) In terms of diversity, the quality of text generated by RANTEXT is superior to that of CUSTEXT+ and SANTEXT+. This phenomenon can be attributed to the design of the *random adjacency list* C_r in RANTEXT, which perturbs tokens to the more probable new ones without retaining them. However, in some specific topics, the variety of tokens is not particularly rich. Additionally, RANTEXT discards proper nouns (those not belonging to V) for privacy protection. As a result, RANTEXT's performance is slightly inferior to that of CUSTEXT+ with respect to MAUVE. (3) From the perspective of coherence, experimental results indicate that RANTEXT and CUSTEXT+ outperform SANTEXT+. This is likely because SANTEXT+ uses the entire vocabulary as its *static adjacency list*, which is too large for the utility of the perturbed text.

For practical deployment, we measured the time cost per inference in `InferDPT`. As illustrated in Table VI, experimental results indicate that `InferDPT` does not require a significant amount of time. Most of the additional time is spent in the extraction module, which is less than 4 seconds.

We also investigated whether `InferDPT` works in different local models. Table VII demonstrate that `InferDPT` works well with different models and various privacy parameter ε. We further compared the cosine similarity between the final generation of `InferDPT` and the output generated by GPT-4 without any privacy protection. The result of the comparison is depicted in Table VIII. Under the same privacy parameter ε across three datasets, the perturbed generation of RANTEXT generally exhibits cosine similarity values close to that of the best-performing CUSTEXT+. This is likely because RANTEXT

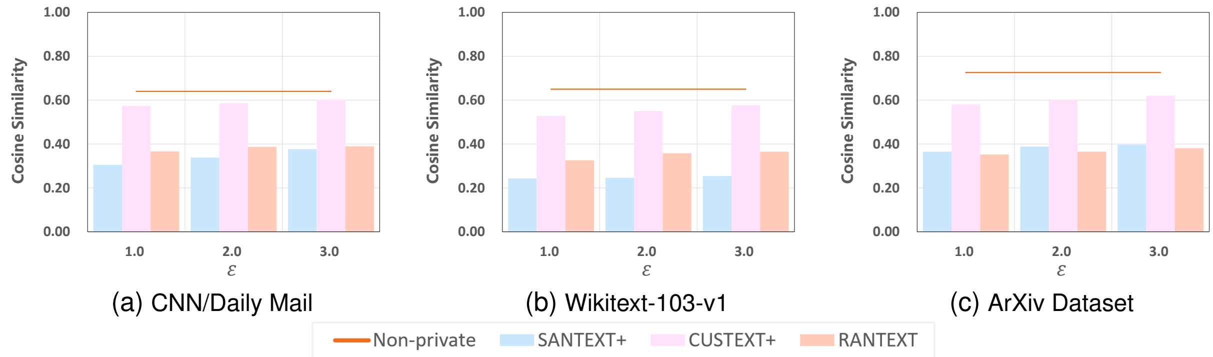

Fig. 9. Cosine similarity ↓ between the perturbed generation and the raw document.

TABLE IX
PRIVACY LEAKAGE VIA THE PERTURBED GENERATION.

ε	Method	Privacy Leakage Rate↓		
		1-gram	2-gram	3-gram
∞	Non-private	0.110	0.058	0.027
2.0	SANTEXT+	0.090	0.030	0.016
	CUSTEXT+	0.098	0.030	0.007
	RANTEXT	**0.078**	**0.024**	**0.004**
6.0	SANTEXT+	0.103	0.053	0.022
	CUSTEXT+	0.099	0.034	0.008
	RANTEXT	**0.081**	**0.027**	**0.005**
10.0	SANTEXT+	0.104	0.055	0.022
	CUSTEXT+	0.103	0.04	0.013
	RANTEXT	**0.095**	**0.029**	**0.006**
14.0	SANTEXT+	0.105	0.057	0.023
	CUSTEXT+	0.105	0.052	0.017
	RANTEXT	**0.101**	**0.032**	**0.008**

discards proper nouns (those not belonging to V) in Doc, whereas CUSTEXT+ retains all of this key information without perturbation. This phenomenon is particularly evident in the Wikitext-103-v1 and ArXiv datasets, which contain more proper nouns. We emphasize that this is also one of the reasons why CUSTEXT+ is vulnerable to the embedding inversion attack.

Furthermore, we investigated the impact of the privacy parameter ε on the probability distribution of the size of the *random adjacency list* C_r in RANTEXT. We use C_r/V to represent the proportion of C_r in the entire vocabulary V. As shown in Figure 6, the *random adjacency list* of RANTEXT is generally larger than the *static adjacency list*t in CUSTEXT+ and smaller than that in SANTEXT+, which provides a better balance between utility and privacy protection of perturbation.

VII. Defense against Privacy Threat

A. BERT inference attack

In the *BERT inference attack* [11], an adversary employs a pre-trained BERT model to recover raw document Doc from their perturbed version Doc_p. The BERT model, developed through masked language modeling [40], predicts the raw tokens by sequentially replacing each token in the perturbed text with a special token "`[MASK]`". This approach leverages BERT's capability to understand context, allowing it to infer the masked tokens. An attack is successful if the output token matches the input token. Subsequently, we calculate attack success rate[3] across all attacks, denoted as r_{ats}. The privacy protection level of the differentially private mechanism is defined as $1 - r_{ats}$.

As shown in Figure 7, RANTEXT offers better privacy protection against *BERT inference attack* compared to SANTEXT+ and CUSTEXT+. The experimental results indicate that RANTEXT provides over 80% privacy protection within an ε value range of 0.01 to 18.0. In particular, with an ε value of 18.0 on the CNN/Daily Mail dataset, the privacy protection level of RANTEXT is 1.11 $\times$ that of SANTEXT+ and 1.41 $\times$ that of CUSTEXT+. We analyzed the results of the experiment and found that BERT did not recognize the tokens of GPT-4. To more comprehensively evaluate RANTEXT's security, we proposed an adaptive attack leveraging the capabilities of GPT-4 in Section VII-C, GPT inference attack.

B. Embedding Inversion Attack

Embedding inversion attack [16] computes the distance between the embedding of each token in the perturbed document and the embeddings of other tokens in the vocabulary, returning top K tokens with the closest Euclidean distance. The privacy protection level is defined as $1 - r_{ats}$.

Experiments were conducted under the conditions of top $K = 1$ and 10. Figure 8 illustrates that, under both conditions, SANTEXT+ and CUSTEXT+ are susceptible to embedding inversion attacks, indicating a relatively lower level of privacy protection. Even at $\varepsilon = 0.01$, these methods could only provide privacy protection for over 40% of the original documents. As the top K changes from 1 to 10, the privacy protection level of SANTEXT+ and CUSTEXT+ remains largely unchanged. On the other hand, RANTEXT benefits from its design of the *random adjacency list* (generally larger than that in CUSTEXT+) and the perturbation on each token, preventing attackers from successfully reconstructing raw tokens.

C. Adaptive Attack: GPT Inference Attack

RANTEXT applies perturbations to the GPT-4 token vocabulary. Since GPT-4 recognizes all tokens, it is hypothesized

[3]We exclude "`[UNK]`" token, as it does not yield meaningful information.

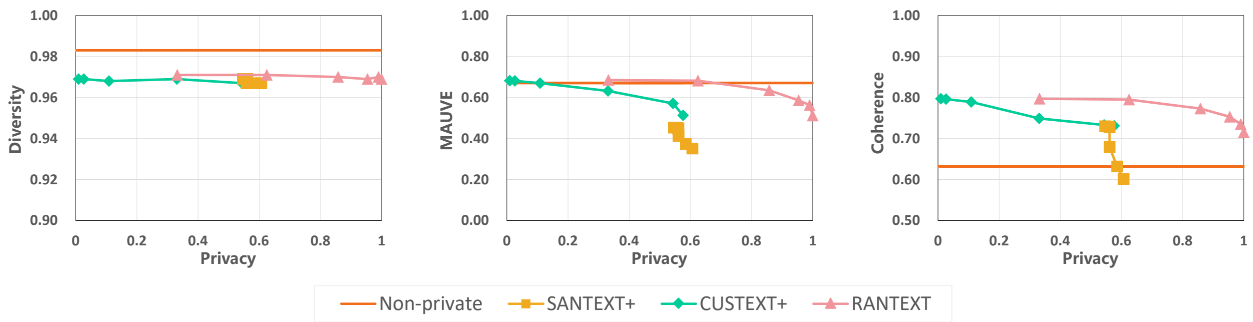

Fig. 10. Results of the trade-off between utility and privacy protection with various privacy parameters ε ranging from 0.01 to 18.0.

that GPT-4 can better reconstruct raw tokens perturbed by RANTEXT. Therefore, we propose an adaptive attack, the *GPT inference attack*. In this method, the attacker inputs perturbed text into GPT-4 and instructs it to recover each token. The attack is successful if the recovered token coincides with the raw one. The privacy protection level is defined as $1 - r_{ats}$. The prompt of this attack can be found in Appendix D.

Figure 7 displays the results of the GPT inference attack. GPT-4 has a higher attack success rate than BERT in all tests. This may be due to GPT-4's larger size and better understanding abilities, making it more effective in inference attacks. Confronted with the GPT inference attack, SANTEXT+ and CUSTEXT+ showed lower privacy protection levels than RANTEXT, which maintained the best privacy protection.

D. Privacy Leakage in Perturbed Generation

We further discussed the possibility of the raw document Doc being leaked by the perturbed generation result Gen_p. Figure 9 shows the cosine similarity between Doc and Gen_p. The orange straight line indicates the cosine similarity between Doc and the generation of GPT-4 without any privacy protection. Experimental results reveal that RANTEXT maintains low semantic similarity between the raw document Doc and the perturbed generation result Gen_p, indicating the low risk of privacy leakage through perturbed results.

Moreover, we measured privacy leakage in perturbed outputs by checking if n-gram tokens from the original document were repeated. A n-gram token found in both raw text and perturbed output counts as a leak. As Table IX shows, even with non-private prompts, under 11% privacy of raw document is leaked.

E. Trade-off between Privacy and Utility

In this subsection, we compare RANTEXT with CUSTEXT+ and SANTEXT+ in terms of privacy-utility trade-offs. we conduct experiments on the CNN/Daily Mail dataset using Vicuna-7b-4bit(3.89GB) as the extraction module. As shown in Figure 10, each point represents the privacy protection level (under top-1 embedding inversion attack) and generation quality of a specific perturbation mechanism and a ε value. The yellow straight line indicates the generation quality of directly using GPT-4 without any privacy protection, referred to as '*non-private*'. The experimental results demonstrate that RANTEXT tends to offer the best generation quality under the same privacy protection level compared to baseline methods. Due to the effectiveness of extraction, the coherence of `InferDPT` is higher than that of non-private GPT-4 in most cases. (More experiments are in Appendix E.)

In summary, RANTEXT demonstrates superior privacy protection against various attacks on differentially private mechanisms compared to baselines, confirming its robust privacy safeguarding alongside high-quality text generation.

VIII. Discussion and Limitations

A. Performance Gap in MAUVE

Although experimental results demonstrate the effectiveness of `InferDPT` in privacy-preserving text generation, a notable gap in MAUVE scores persists when compared to GPT-4, as shown in Table VII. One probable reason for this discrepancy is the semantic perturbations introduced by LDP, which disturb the original information in the raw prompt. Future work focusing on developing a differentially private mechanism with a better trade-off between utility and privacy protection could improve the MAUVE score.

Furthermore, it is important to note that the local model within the extraction module of `InferDPT` is pre-trained and not specifically fine-tuned for this task. Future work that enhances the extraction and reconstruction capabilities of this local model could also result in improved MAUVE scores.

B. Comparing to Prompt Engineering Methods

The prompt engineering method, represented by HaS [41], trains a language model to identify private entities and randomly replace them with new words sampled by another language model. Experimental results demonstrate its effectiveness for privacy-preserving classification and translation tasks. However, the perturbed text in HaS is not required to be semantically relevant to the raw text. The unconstrained perturbation makes it unsuitable for open-ended text generation tasks, as its significant semantic bias could lead to semantically irrelevant generations. Furthermore, it only protects specific words of private entities, leaving others (that are not detected by HaS) exposed to adversaries.

DP-Prompt [42] leverages the power of pre-trained LLMs and zero-shot prompting to counter author de-anonymization attacks [43] while minimizing the impact on downstream utility. It provides LDP-based privacy protection for classification tasks specifically against de-anonymization attacks. However, the demo[4] of DP-Prompt reveals the privacy leakage of personally identifiable information (PII) from its sentence-level perturbations. More importantly, it does not address the information bias introduced by the LDP, thus rendering it unsuitable for text generation tasks.

Compared to existing HaS, our proposed `InferDPT` utilizes LDP to replace the raw token with a randomly selected one that is close in embedding distance, thereby maintaining the utility of the perturbed text. To address the semantic bias that DP-Prompt does not solve, `InferDPT` locally deploys a small language model to generate an aligned output with the input of the perturbed generation and the raw document.

C. Limitations of InferDPT

The framework for privacy-preserving text generation presented in this paper has two main limitations.

First, `InferDPT` requires the deployment of a small language model in its extraction module. In scenarios with extremely limited computational resources (e.g., smartwatch [44]), this requirement might not be feasible.

Second, there exists a gap in MAUVE scores between `InferDPT` and direct usage of GPT-4. There is still room for proposing a new differentially private mechanism with a better balance between utility and privacy protection to enhance the generation results in this metric.

D. Privacy Budget of InferDPT

As previously mentioned, the perturbation module of `InferDPT` generates a perturbed document by replacing each token in the raw document with a new one sampled using local differential privacy (LDP). It is important to note that each token in the raw document undergoes the LDP process only once. Therefore, the token-level perturbation [11], [12] in `InferDPT` introduces no accumulated privacy risks. For instance, when using RANTEXT (which satisfies ε-LDP for its sampling process) as the perturbation module, the privacy budget for each raw token in `InferDPT` remains ε.

IX. Related Work

Secure two-party inference. Iron [45] and CipherGPT [8] have applied homomorphic encryption [46] to language models that are based on Transformer [47]. They perform inference on encrypted data. However, it results in a problem that cannot be completely solved today: the significant computation time and communication costs. Taking CipherGPT as an example, it infers a token costing 24 minutes and 93 GB of bandwidth, making the deployment of encrypted inference impractical.

Privacy-preserving prompt learning (tuning). Prompt-PATE [48] and DP-OPT [49] have utilized differential privacy (DP) to reconstruct the datasets used for classification tasks, thereby protecting the privacy of training data during the prompt learning (tuning) process [50]. However, these methods do not protect the private data of users in the prompt during the inference process with LLMs. Also, they focus on the classification tasks and do not solve the information distortion introduced by the noise of differentially private mechanisms.

Privacy-preserving in-context learning. Tang et al. [51] introduced a differentially private approach to generate privacy-preserving examples for in-context learning [52]. They deploy a large language model to reconstruct the private examples via the few-shot generation of differential privacy. They also focus on the classification task and do not protect the input document during the inference process of generation tasks.

Privacy-preserving model training. SANTEXT+ [11] and CUSTEXT+ [12] have utilized differential privacy to enhance text privacy. They sequentially substitute words in texts with semantically similar words to preserve privacy during training in classification tasks. These two mechanisms are resistant to the input inference attack [11]. However, they are vulnerable to the embedding inversion attacks [16]. They do not solve semantic distortion caused by DP noise. They are unsuitable for direct use in text generation tasks.

X. Conclusion

This paper explores the challenge of privacy leakage in text generation tasks executed by black-box large language models and introduces `InferDPT` as a potential solution. Additionally, we propose RANTEXT, a novel differential privacy algorithm designed for large language models following the exponential mechanism to enhance user privacy protection. We expect that our solution and findings can provide technical insights into the current privacy challenges and shed light on potential future explorations in privacy protection within emerging LLMs.

References

[1] D. Li, A. S. Rawat, M. Zaheer, X. Wang, M. Lukasik, A. Veit, F. Yu, and S. Kumar, "Large language models with controllable working memory," in *Findings of the Association for Computational Linguistics: ACL 2023* (A. Rogers, J. Boyd-Graber, and N. Okazaki, eds.), (Toronto, Canada), pp. 1774–1793, Association for Computational Linguistics, July 2023.

[2] "Chatgpt, a model which interacts in a conversational way." https://chat.openai.com/?model=gpt-4-browsing, 2023.

[3] "Openai's chatgpt now has 100 million weekly active users." https://techcrunch.com/2023/11/06/openais-chatgpt-now-has-100-million-weekly-active-users/, 2023.

[4] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[5] "Samsung employees leak proprietary information via chatgpt." https://www.cshub.com/data/news/iotw-samsung-employees-allegedly-leak-proprietary-information-via-chatgpt, 2023.

[6] "Gpt-3.5 models can understand and generate natural language or code." https://chat.openai.com/?model=text-davinci-002-render-sha, 2023.

[7] "Gpt-3.5 leaked a random individual's photo in the output." https://twitter.com/thealexker/status/1719896871009694057, 2023.

[8] X. Hou, J. Liu, J. Li, Y. Li, W.-j. Lu, C. Hong, and K. Ren, "Ciphergpt: Secure two-party gpt inference," *Cryptology ePrint Archive*, 2023.

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

[10] M. Du, X. Yue, S. S. Chow, T. Wang, C. Huang, and H. Sun, "Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass," *arXiv preprint arXiv:2309.06746*, 2023.

[4] https://github.com/SaitejaUtpala/dp_prompt/blob/main/data/chatgpt_data/chatgpt_zero_shot_paraphrase_imdb.zip

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

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

[13] "Gpt-4 is openai's most advanced system, producing safer and more useful responses." https://openai.com/gpt-4, 2023.

[14] C. Dwork, "Differential privacy," in *International colloquium on automata, languages, and programming*, pp. 1–12, Springer, 2006.

[15] Y. Zhu, S. Lu, L. Zheng, J. Guo, W. Zhang, J. Wang, and Y. Yu, "Texygen: A benchmarking platform for text generation models," in *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 1097–1100, 2018.

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

[17] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[18] S. Kotz, T. Kozubowski, and K. Podgórski, *The Laplace distribution and generalizations: a revisit with applications to communications, economics, engineering, and finance*. No. 183, Springer Science & Business Media, 2001.

[19] "Claude is a next generation ai assistant built for work and trained to be safe, accurate, and secure.." https://claude.ai/, 2023.

[20] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[21] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. E. Gonzalez, *et al.*, "Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality," *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2023.

[22] S. Welleck, I. Kulikov, S. Roller, E. Dinan, K. Cho, and J. Weston, "Neural text generation with unlikelihood training," *arXiv preprint arXiv:1908.04319*, 2019.

[23] S. Sharma, T. Nayak, A. Bose, A. K. Meena, K. Dasgupta, N. Ganguly, and P. Goyal, "Finred: A dataset for relation extraction in financial domain," in *Companion Proceedings of the Web Conference 2022*, pp. 595–597, 2022.

[24] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith, "What can we learn privately?," *SIAM Journal on Computing*, vol. 40, no. 3, pp. 793–826, 2011.

[25] F. McSherry and K. Talwar, "Mechanism design via differential privacy," in *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pp. 94–103, 2007.

[26] M. Alvim, K. Chatzikokolakis, C. Palamidessi, and A. Pazii, "Local differential privacy on metric spaces: optimizing the trade-off with utility," in *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pp. 262–267, IEEE, 2018.

[27] N. L. Toolkit. https://www.nltk.org, 2023.

[28] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," *Advances in neural information processing systems*, vol. 28, 2015.

[29] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[30] S. Merity, C. Xiong, J. Bradbury, and R. Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016.

[31] C. B. Clement, M. Bierbaum, K. P. O'Keeffe, and A. A. Alemi, "On the use of arxiv as a dataset," 2019.

[32] J. Xu, X. Liu, J. Yan, D. Cai, H. Li, and J. Li, "Learning to break the loop: Analyzing and mitigating repetitions for neural text generation," *Advances in Neural Information Processing Systems*, vol. 35, pp. 3082–3095, 2022.

[33] K. Lagler, M. Schindelegger, J. Böhm, H. Krásná, and T. Nilsson, "Gpt2: Empirical slant delay model for radio space geodetic techniques," *Geophysical research letters*, vol. 40, no. 6, pp. 1069–1073, 2013.

[34] X. Lin, S. Han, and S. Joty, "Straight to the gradient: Learning to use novel tokens for neural text generation," in *International Conference on Machine Learning*, pp. 6642–6653, PMLR, 2021.

[35] K. Pillutla, S. Swayamdipta, R. Zellers, J. Thickstun, S. Welleck, Y. Choi, and Z. Harchaoui, "Mauve: Measuring the gap between neural text and human text using divergence frontiers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 4816–4828, 2021.

[36] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," *arXiv preprint arXiv:2104.08821*, 2021.

[37] "Fast bpe tokeniser for use with openai's models." https://github.com/openai/tiktoken, 2023.

[38] "New and improved embedding model." https://openai.com/blog/new-and-improved-embedding-model, 2023.

[39] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "Gptq: Accurate post-training quantization for generative pre-trained transformers," *arXiv preprint arXiv:2210.17323*, 2022.

[40] H. Bao, L. Dong, F. Wei, W. Wang, N. Yang, X. Liu, Y. Wang, J. Gao, S. Piao, M. Zhou, *et al.*, "Unilmv2: Pseudo-masked language models for unified language model pre-training," in *International conference on machine learning*, pp. 642–652, PMLR, 2020.

[41] Y. Chen, T. Li, H. Liu, and Y. Yu, "Hide and seek (has): A lightweight framework for prompt privacy protection," *arXiv preprint arXiv:2309.03057*, 2023.

[42] S. Utpala, S. Hooker, and P. Chen, "Locally differentially private document generation using zero shot prompting," in *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023* (H. Bouamor, J. Pino, and K. Bali, eds.), pp. 8442–8457, Association for Computational Linguistics, 2023.

[43] J. Bevendorff, M. Potthast, M. Hagen, and B. Stein, "Heuristic authorship obfuscation," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 1098–1108, 2019.

[44] X. Chen, T. Grossman, D. J. Wigdor, and G. Fitzmaurice, "Duet: exploring joint interactions on a smart phone and a smart watch," in *Proceedings of the SIGCHI Conference on human factors in computing systems*, pp. 159–168, 2014.

[45] M. Hao, H. Li, H. Chen, P. Xing, G. Xu, and T. Zhang, "Iron: Private inference on transformers," *Advances in Neural Information Processing Systems*, vol. 35, pp. 15718–15731, 2022.

[46] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169–178, 2009.

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[48] H. Duan, A. Dziedzic, N. Papernot, and F. Boenisch, "Flocks of stochastic parrots: Differentially private prompt learning for large language models," *arXiv preprint arXiv:2305.15594*, 2023.

[49] J. Hong, J. T. Wang, C. Zhang, Z. Li, B. Li, and Z. Wang, "Dp-opt: Make large language model your privacy-preserving prompt engineer," *arXiv preprint arXiv:2312.03724*, 2023.

[50] N. Ding, S. Hu, W. Zhao, Y. Chen, Z. Liu, H.-T. Zheng, and M. Sun, "Openprompt: An open-source framework for prompt-learning," *arXiv preprint arXiv:2111.01998*, 2021.

[51] X. Tang, R. Shin, H. A. Inan, A. Manoel, F. Mireshghallah, Z. Lin, S. Gopi, J. Kulkarni, and R. Sim, "Privacy-preserving in-context learning with differentially private few-shot generation," *arXiv preprint arXiv:2309.11765*, 2023.

[52] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.