Generate Synthetic Text Approximating the Private Distribution with Differential Privacy

Wenhao Zhao, Shaoyang Song, Chunlai Zhou *
Computer Science Dept, Renmin University of China, Beijing, CHINA
{zhaowh, songshaoyang, czhou}@ruc.edu.cn

Abstract
Due to the potential leakage of sensitive information in text, there is a societal call for feeding privacy-preserving text to model training. Recently, a lot of work showed that using synthetic text with differential privacy, rather than private text, can provide a strong privacy protection. However, achieving higher semantic similarity between synthetic and private text has not been thoroughly investigated. In this paper, we propose an approach that combines the iteratively optimized mindset from genetic algorithms to align the distribution of synthetic text with that of private text. Furthermore, not only does the final synthetic text meet the requirements of privacy protection, but also has a high level of quality. Through comparisons with various baselines on different datasets, we demonstrate that our synthetic text can closely match the utility of private text, while providing privacy protection standards robust enough to resist membership inference attacks from malicious users.

1 Introduction
Natural language text can serve not only as training data for natural language processing tasks, for example sentiment analysis, but also as demonstrations in prompts for large language models to enhance their predictive capabilities. However, text often contains sensitive information such as passwords and names, which can lead to potential privacy leakages. To protect sensitive information, the simplest method [Pilán et al., 2022] is to identify the sensitive information and replace it with other words. However, attackers can still identify a user’s identity through statistical information in the text [Narayanan and Shmatikov, 2008], such as catchphrase.

Considering the ability of providing personalized privacy protection settings to balance the trade-off between privacy and data utility, handling sensitive data with differential privacy (DP) has become a golden standard. Text sanitization [Yue et al., 2021; Chen et al., 2023] replace all tokens in the original text with a new token to achieve the privacy guarantee. While differential privacy ensures that this method can resist attacks at the token level (e.g. mask token inference attack), such token-level private mechanism is unable to provide effective privacy protection against a broader range of attack methods. This is because text sanitization cannot change the structure of the text and attackers often have a significant chance of illegally obtaining private text information through membership inference attacks (MIA) [Shokri et al., 2017; Carlini et al., 2021].

Recently, generating differentially private synthetic text for downstream tasks is gradually becoming a common practice. Figure 1 illustrates that applying synthetic text as demonstrations in prompt can effectively protect private datasets. To obtain synthetic text, one approach involves training large models using differential privacy [Abadi et al., 2016; Anil et al., 2021; Yue et al., 2022]. These methods primarily focus on adding calibrated noise to gradients or text representations during the training phase to prevent the inference of sensitive user data from the trained language models. However, this approach requires significant computational resources during training, and when privacy protection parameters are modified, the model needs to be retrained. Another
The fundamental idea of Differential Privacy (DP) [Dwork et al., 2006] is to design a randomized algorithm \( M : D \rightarrow S \). For all neighboring datasets \( D, D' \) (\( D \) and \( D' \) only differ in a single sample) and any set \( S \):
\[
\Pr[M(D) \in S] \leq e^\epsilon \Pr[M(D') \in S] + \delta,
\]
we say the mechanism \( M \) satisfies \((\epsilon, \delta)\)-differential privacy. A significant property of differential privacy is its resilience to post-processing. It ensures differential private outputs to apply arbitrary, data-independent transformations without compromising their privacy guarantees. In our work, this property ensures that synthetic text will not incur additional privacy loss when used for downstream tasks.

### 2.2 Privacy-preserving Text Embeddings

Many text encoders [Devlin et al., 2018; Ni et al., 2021] have the capability to represent a natural language sentence in the form of a high-dimensional embedding. However, an increasing amount of research demonstrates that embeddings are likely to leak information about the original text [Song and Raghunathan, 2020; Pan et al., 2020; Li et al., 2023].

In order to prevent untrusted servers from extracting sensitive information from text, one approach [Du et al., 2023a; Du et al., 2023b] is to sanitize text embeddings to ensure differential privacy, and then send them to the server for fine-tuning on downstream tasks. Specifically, [Du et al., 2023b] propose DP-forward which directly perturbs embedding matrices in the forward pass of text encoders. However, being able to provide only embedded information will face limitations in terms of applicability to downstream tasks. For example, the input must be textual information for in-context learning task.

Another approach [Meehan et al., 2022; Lin et al., 2023] is to take the advantage of public data. [Meehan et al., 2022] firstly sample a set of non-private text from public data. After mapping both public texts and private texts through the same text encoder, they select public embeddings that near to the private embedding distribution center with exponential mechanism (EM) [McSherry and Talwar, 2007]. This approach essentially involves selecting a portion from non-private public data as privacy-preserving embeddings, and the performance of downstream tasks largely depends on the distribution of the public data.

### 2.3 Inversion from Embedding to Text

For a certain text encoder \( \varphi \), we attempt to recover the original text \( x \) based on its embedding \( e = \varphi(x) \). Because a text encoder requires the embeddings of semantically similar texts to ideally be similar, this provides us with insights into the process of inverting embeddings into text. Specifically, the training process of the text decoder in [Morris et al., 2023] involves iteratively self-correcting [Welleck et al., 2022] the recovered text, achieving a gradual convergence of the embeddings between the recovered text and the original text:

\[
p(x^{(t+1)}|e) = \sum_{x^{(t)}} p(x^{(t)}|e)p(x^{(t+1)}|e, x^{(t)}, \varphi(x^{(t)})),
\]

where \( x^{(t+1)} \) represents the correction of \( x^{(t)} \) and \( x^{0} \) is the initial hypothesis generation. In our work, we need to train a text decoder to invert from privacy-preserving embeddings to synthetic text.

### 3 Method

We aim to generate synthetic text that satisfies the following three requirements:

**Requirement 1.** The leakage of sensitive information from synthetic text must be within a controllable range.

**Requirement 2.** Synthetic text should have high readability.

**Requirement 3.** The distribution of the synthetic text should approximate the distribution of private text.

### 3.1 Synthetic Text Generation

**Preparation:** Train Text Decoder on Public Texts. To measure the difference between synthetic text distribution and private text distribution, the embedding distance is a common metric. Corresponding to the text encoder, we need to train a decoder to restore the embeddings back to text. Following
Figure 2: Overview of our method. We approximate the private text distribution by iteratively updating synthetic distribution. Parent selection is the only step that involves access to private text. $Syn^t$ represents the embedding distribution of the population at iteration $t$.

previous work [Morris et al., 2023], we use non-private public texts to train the Decoder while freezing all the pretrained parameters of text encoder. It is important to note that, as all training processes are conducted on public datasets, there is no consumption of the privacy budget in this stage. Furthermore, the decoder in the preparation stage only needs to be trained once and later can be reused to generate synthetic text for different private datasets. Therefore, we consider the computational resource cost to be acceptable.

Overview The most intuitive way to protect privacy is to add noise to embeddings, but it leads to extremely poor text readability after decoding. This is because text encoders typically generate high-dimensional text embeddings to convey the abundant semantic information in the text. Protecting high-dimensional vectors requires huge noise, often causing the noisy points to no longer reside in a space enable to successful decoding. To address this issue, we propose a novel framework (Figure. 2) for generating synthetic text, with iteratively approaching the distribution of private text (meet Requirement 3.) within the successfully decoded space (meet Requirement 2.). Simultaneously, during each iteration, we apply a LimitedDomain mechanism [Durfee and Rogers, 2019] (meet Requirement 1.) to protect the privacy of the private text. Next, we will discuss three important components in our method and more details are presented in Algorithm 1.

- initial population: Our initial population $E_{pb}$ is defined as embeddings of public text, where public text can be a set of texts related to private text datasets collected from the internet. For example, if the private text is about movie reviews, public data can be selected from publicly available movie review sections online. Another simpler way to obtain public data is to generate text with appropriate instructions using zero-shot prompting.

- private selection: The fitness score of an individual in the population is determined by the cosine distance between its embedding and the embeddings in private datasets.

The more private samples are similar to one synthetic sample, the more likely that the synthetic sample is to be selected as a parent for the next generation. Because of the exposure to private data during the selection process, we must add noise to ensure differential privacy.

- offspring generation: Similar to genetic algorithms, estimation of distribution algorithms (EDA) primarily employs probabilistic models and sampling in an implicit form to generate new individuals. In our work, we utilize the Gaussian distribution model to assess the probability distribution of the population [Wang et al., 2020; Mitchell and Taylor, 1999]. Due to the randomness in sampling the offspring, EDA preserves high diversity and strong global search ability.

Private Selection. As the size of offspring population $N_{cld}$ increases, the higher chance of individuals in the next generation can be more similar to private texts. However, a large size of offspring population can also flatten the neighbour histogram. When we apply privacy protection mechanisms (e.g., Gaussian mechanism) into the histogram, the noise often plays a crucial role during the selection of parent samples, potentially significantly impacting the convergence speed of the synthetic distribution. In our work, we apply the LimitedDomain mechanism to narrow down the selection range from $N_{cld}$ to $K$, where these $K$ samples are the ones with the highest vote count in the histogram without DP-noise. Then, we select up to $N_{par}$ samples from the histogram with DP-noise as parent samples. It should be noted that the LimitedDomain algorithm does not guarantee the output of exactly $N_{par}$ indices. When each individual has a roughly equal amount of private text that is most similar to it, in order to preserve privacy, LimitedDomain mechanism may output the empty set. In that case, we believe that our synthetic distribution is close enough to the private distribution.

Due to the inherent randomness in generating the next generation, allowing undecodable samples to continue as parent
samples is likely to result in the embedding space of the population increasingly diverging from the successful decoding space as the iterations progress. Although it remains to see whether low-perplexity texts are more effective in all cases [Gonen et al., 2022; Shin et al., 2022], we further filter the selected \( N_{par} \) embeddings with synthetic text perplexity check operation to better demonstrate the utility of our method. Specifically, after inverting embeddings back to text through the pretrained \( \text{Decoder} \) in the preparation stage, we use the perplexity of the text as a measure of text readability. If the perplexity exceeds the predefined threshold \( H \), we will remove the corresponding embedding from the parent set \( E_{syn} \).

Algorithm 1: Synthetic Text Generation

Input:
1. private embeddings: \( E_{pr} = \{e_{pr}^i\}_{i=1}^{N_{pr}} \)
2. public embeddings: \( E_{pb} = \{e_{pb}^i\}_{i=1}^{N_{pb}} \)
3. size of parent set: \( N_{par} \)
4. size of offspring population: \( N_{cld} \)
5. number of iteration: \( T \)
6. size of limited domain: \( K \)
7. threshold for synthetic text readability: \( H \)

Output: synthetic text set: \( S_{syn} \)

\begin{algorithmic}
1. \( E_{pop} \leftarrow E_{pb} \)
2. \( E_{syn}, S_{syn} \leftarrow \{\}, \{\} \)
3. for \( t \leftarrow 1 \ldots T \) do
   4. \# find similar samples
      5. \( \text{hist}_t[0] \leftarrow [0], [0] \)
      6. for \( e_{pr} \) in \( E_{pr} \) do
      7. \# ensure differential privacy
      8. \( j = \text{argmin}_j \leq \text{len}(e_{pop}) \cdot \cos(e_{pr}^j, e_{gen}^j) \)
      9. \( \text{hist}_t[j] = \text{hist}_t[j] + 1 \)
    end
   10. \# filter low-readability text
    11. for \( id \) in \( \text{rank}_{dp} \) do
    12. \# generate next population with EDA
        13. if \( \text{Perplexity}(text_{syn}) < H \) then
           14. \( E_{syn} \cup E_{pop}[id] \)
           15. \( S_{syn} \cup \text{text}_{syn} \)
        end
    16. end
    17. end
    18. end
   19. return \( S_{syn} \)
\end{algorithmic}

Offspring Generation with EDA. Firstly, we build a Gaussian probability distribution model based on individuals in the current population. In order to make the synthetic embedding distribution more likely to get closer to the private embedding distribution after one round of iteration, we employ a smooth approach to update the synthetic distribution. Specifically, given a smoothing parameter \( \alpha \), we move towards the direction of the optimal individual and the suboptimal individual, while simultaneously moving away from the worst individual (line 3 in Algorithm 2). The variation of the new distribution is determined collectively by the top-\( R \) individuals (line 4 in Algorithm 2). Finally, individuals for the next iterations are sampled based on the new distribution with a mean value of \( \mu \) and a variance value of \( \sigma \).

Algorithm 2: Estimate Distribution Algorithm

Input:
1. current population embeddings: \( E = \{e_i\}_{i=1}^N \)
2. fitness score rank (descending order): \( \text{rank} \)
3. next population size: \( N_{cld} \)
4. smooth parameter: \( \alpha \)
5. the number of sample size: \( R \)

Output: next population embeddings: \( \hat{E} = \{\hat{e}_i\}_{i=1}^{N_{cld}} \)

\begin{algorithmic}
1. \( \hat{e}_i, \hat{e} \leftarrow \frac{\text{len} - 1}{N} \cdot e_i \cdot \frac{N - 1}{\text{len} - 1} \cdot e_i \)
2. \( \sigma = \sqrt{\frac{\text{len} - 1}{N} \cdot (e_i - \mu)^2} \)
3. \( \mu = (1 - \alpha) \cdot \mu + \alpha \cdot (e_{\text{rank}[0]} + e_{\text{rank}[1]} - e_{\text{rank}[0]} - e_{\text{rank}[1]}) \)
4. \( \hat{\sigma} = (1 - \alpha) \cdot \sigma + \alpha \cdot \sqrt{\frac{(\text{len} - 1)(e_{\text{rank}[0]} - e_{\text{rank}[1]})}{R}} \)
5. Repeat \( N_{cld} \) times: \( \hat{e}_i \sim N(\mu, \hat{\sigma}) \)
6. return \( \hat{E} \)
\end{algorithmic}

3.2 Privacy Analysis

Theorem 1. Exponential Mechanism satisfies \( \epsilon \)-DP.

In exponential mechanism, defining the scoring function \( q(D, y) \) is crucial, where \( q(D, y) \) represents the evaluation of \( y \)'s performance on dataset \( D \). In our work, the scoring function can also be regarded as the concept of the fitness function in genetic algorithms. Specifically, we define the score function by the number of most similar neighbors in the private dataset \( D \) corresponding to a particular individual sample \( y \) in the current population.

Theorem 2. Alg. 3 satisfies \( (\epsilon', \delta + \delta') \)-DP where

\[
\epsilon' = \min \left\{ \frac{k\epsilon}{2}, \frac{k\epsilon}{2} \cdot \left( \frac{e_i}{e_j} + 1 \right) + \epsilon \sqrt{2k \ln 1/\delta'}, \frac{k\epsilon}{2} \right\}.
\]

The proof is derived from [Durfee and Rogers, 2019], where it essentially represents an exponential mechanism. Applying Gumbel noise and simultaneously selecting the top-k as parent samples is equivalent to applying the exponential mechanism to select the top-1 sample, followed by the removal of that index and iterative processing. The privacy cost associated with restricting the domain size is incorporated into the \( \delta \) term.

Theorem 3. If we set the privacy parameter in LimitedDomain as \( \epsilon_0, \delta_0 \), the total privacy bound of our DP algorithm in
We assume text from the following three datasets are considered as private text that needs to be protected:

- **AGNews** dataset [Zhang et al., 2015] consists of approximately 120,000 news articles categorized into four classes: World, Sports, Business, and Science/Technology.
- **Disaster** dataset [Bansal et al., 2019] originate from news reports or Twitter, with 4342 samples describing different disasters (e.g., fire, flood), while an additional 3271 samples could mention about any topic other than disasters.
- **Trec** [Voorhees and Tice, 2000] dataset comprises questions from 6 different categories, such as numbers, locations, etc. The distribution of the 5500 questions in the training set and the 500 questions in the test set is uneven across these 6 question labels.

### 4.1 Datasets

4.2 Experiment Setup

- In initial population step: we select 1000 public texts as our initial population and GTR-base [Ni et al., 2021] model to embed public texts and private texts.
- In private selection step: we follow the common practice to set $\delta = 1/|D|$ where $|D|$ is the size of private dataset.

The domain size of being able to become a parent sample is 300, and 30 samples are selected from them. GPT-2 model [Brown et al., 2020] is used to obtain the perplexity and filter out texts with perplexity exceeding the threshold of 50.

- In offspring generation step: a large smoothing parameter $\alpha$ will lead to a high degree of homogenization among the final synthetic texts. Therefore, we set $\alpha$ as 0.1 and sample 3000 samples from the updated distribution for the next iteration.

### 4.3 Baselines

We compare the performance of our method with 3 baselines: *CusText* [Chen et al., 2023]: for each token in private text, assign a customized set of output tokens and replace the original token with one of the corresponding output tokens based on the EM mechanism.

- **DP-ICL** [Tang et al., 2023]: predict the next token across different subsets of private text and add gaussian noise [Dwork et al., 2006] during aggregation. Eventually, all predicted tokens are concatenated together to form a single synthetic text.

- **SentDP** [Meehan et al., 2022]: the Tukey Depth [Tukey, 1975] relative to the private distribution of public texts is designed as score function for EM mechanism, and the selected public texts are used in downstream tasks directly.

### 4.4 Main Results

We use in-context learning task to investigate the performance of synthetic text. We extracted 6 samples with evenly distributed labels from the synthetic text set generated by each method as demonstrations for the prompt. Because of the varying abilities of large language models to extract useful information from context, to demonstrate the applicability of our synthetic text, we conducted experiments with three models of different sizes: babbage (1.3B), curie (6.7B), and davinci (175B).

From Table. 1, our synthetic text surpasses existing baselines in many cases. Compared to the DP-ICL method, each individual in the population can be restored to a synthetic text. However, in DP-ICL, we not only need multiple requests to the large model interface but also the text generation process is token-by-token, making the synthesis of one single text sample time-consuming. Another observation is that the variance of our results is much lower than SentDP. This is because SentDP, lacking an iterative process towards the private distribution, heavily relies on whether the distance between the initial public text and private text distributions is close enough or not. To reduce the variance of SentDP, one feasible approach is to increase the size of the public text set. However, this comes with additional data collection costs.

### 4.5 Ablation Study

**Varying Privacy Budget.** We present the 6-shot in-context learning ability of synthetic texts under different privacy budget conditions in Table. 2. Under all privacy budget settings, the evaluation results on the test set, whether using synthetic text or private text as demonstrations, outperform the zero-shot scenario. Even when the privacy budget is relatively

---

**Algorithm 3: LimitedDomain**

**Input:**
1. neighbour histogram: $hist$
2. privacy parameter: $\epsilon, \delta$
3. size of limited domain: $K$
4. size of selected samples: $k$

**Output:** set of selected indices

1. sort $hist$ in descending order that $h_1 \geq h_2 \geq ...$
2. $h_{\perp} \leftarrow h_{K+1} + 1 + 2 \ln(\min\{\Delta, K, \text{len}(h) - K\})/\delta/\epsilon$
3. $v_{\perp} \leftarrow h_{\perp} + \text{Gumbel}(2\Delta_{\infty}/\epsilon)$
4. for $j \leq K$
5. \quad $v_{(j)} \leftarrow h_{(j)} + \text{Gumbel}(2\Delta_{\infty}/\epsilon)$
6. end
7. Sort $\{v_{(j)}\} \cup v_{\perp}$ and let $v_{i_1}, \ldots, v_{i_{(j)}}, v_{\perp}$ be the sorted list up until $v_{\perp}$
8. return $\{i_1, \ldots, i_{(j)}, \perp\}$ if $j < k$
9. otherwise $\{i_1, \ldots, i_k\}$

$T$ iterations is $(\epsilon, T\delta_0 + \delta')$-DP with $\delta' > 0$ and

$$\epsilon = O(\sqrt{T \log(1/\delta')}\epsilon_0)$$

It follows from the advanced composition theorem of differential privacy [Dwork et al., 2010] that the number of iteration is constrained by the privacy budget. A more detailed experimental result analysis in the discuss section will also confirm this point.

### 4 Experiment

#### 4.1 Datasets

We assume text from the following three datasets are considered as private text that needs to be protected:

- **AGNews** dataset [Zhang et al., 2015] consists of approximately 120,000 news articles categorized into four classes: World, Sports, Business, and Science/Technology.
- **Disaster** dataset [Bansal et al., 2019] originate from news reports or Twitter, with 4342 samples describing different disasters (e.g., fire, flood), while an additional 3271 samples could mention about any topic other than disasters.
- **Trec** [Voorhees and Tice, 2000] dataset comprises questions from 6 different categories, such as numbers, locations, etc. The distribution of the 5500 questions in the training set and the 500 questions in the test set is uneven across these 6 question labels.

#### 4.2 Experiment Setup

- In initial population step: we select 1000 public texts as our initial population and GTR-base [Ni et al., 2021] model to embed public texts and private texts.
- In private selection step: we follow the common practice to set $\delta = 1/|D|$ where $|D|$ is the size of private dataset.

The domain size of being able to become a parent sample is 300, and 30 samples are selected from them. GPT-2 model [Brown et al., 2020] is used to obtain the perplexity and filter out texts with perplexity exceeding the threshold of 50.

- In offspring generation step: a large smoothing parameter $\alpha$ will lead to a high degree of homogenization among the final synthetic texts. Therefore, we set $\alpha$ as 0.1 and sample 3000 samples from the updated distribution for the next iteration.

#### 4.3 Baselines

We compare the performance of our method with 3 baselines: *CusText* [Chen et al., 2023]: for each token in private text, assign a customized set of output tokens and replace the original token with one of the corresponding output tokens based on the EM mechanism.

- **DP-ICL** [Tang et al., 2023]: predict the next token across different subsets of private text and add gaussian noise [Dwork et al., 2006] during aggregation. Eventually, all predicted tokens are concatenated together to form a single synthetic text.

- **SentDP** [Meehan et al., 2022]: the Tukey Depth [Tukey, 1975] relative to the private distribution of public texts is designed as score function for EM mechanism, and the selected public texts are used in downstream tasks directly.

#### 4.4 Main Results

We use in-context learning task to investigate the performance of synthetic text. We extracted 6 samples with evenly distributed labels from the synthetic text set generated by each method as demonstrations for the prompt. Because of the varying abilities of large language models to extract useful information from context, to demonstrate the applicability of our synthetic text, we conducted experiments with three models of different sizes: babbage (1.3B), curie (6.7B), and davinci (175B).

From Table. 1, our synthetic text surpasses existing baselines in many cases. Compared to the DP-ICL method, each individual in the population can be restored to a synthetic text. However, in DP-ICL, we not only need multiple requests to the large model interface but also the text generation process is token-by-token, making the synthesis of one single text sample time-consuming. Another observation is that the variance of our results is much lower than SentDP. This is because SentDP, lacking an iterative process towards the private distribution, heavily relies on whether the distance between the initial public text and private text distributions is close enough or not. To reduce the variance of SentDP, one feasible approach is to increase the size of the public text set. However, this comes with additional data collection costs.

#### 4.5 Ablation Study

**Varying Privacy Budget.** We present the 6-shot in-context learning ability of synthetic texts under different privacy budget conditions in Table. 2. Under all privacy budget settings, the evaluation results on the test set, whether using synthetic text or private text as demonstrations, outperform the zero-shot scenario. Even when the privacy budget is relatively
Table 1: Performance comparison of the 6-shot ICL on the test sets of different datasets with various baselines under medium privacy protection ($\epsilon=5$). We conduct the experiment 10 times with different selected synthetic texts and show the average accuracy (on the left) and variance (on the right) of these 10 experiments.

Table 2: Comparison of average accuracy with baseline methods under different privacy budget conditions. When $\epsilon = 0$, it represents a zero-shot scenario, and when $\epsilon = \infty$, the demonstrations are randomly sampled from the private text.

Table 3: Comparison of average accuracy in the condition of $\epsilon = 5$ under different number of demonstrations in prompt.

4.6 Discussion
What is the degree of similarity between synthetic samples and private samples? To measure the distance between abundant, synthetic text can achieve utility similar to that of private text. Furthermore, we found that our method has a greater advantage when the privacy budget is tight and a balance is achieved between privacy and utility when $\epsilon = 5$. On the contrary, when the privacy budget is sufficient, the performance of the CusText method is very close to the result without privacy protection ($\epsilon = \infty$). However, even with the same privacy budget, the privacy protection provided by token-level method is strictly weaker than others.

**Varying number of shots.** Next, we investigated the in-context learning ability with different numbers of shots. As can be seen in Table 3, the optimal number of shots for achieving the best performance varies across different datasets, primarily depending on the number of label categories in the dataset. For the binary-label Disaster dataset, optimal performance is reached with 6-shot, while for the 6-label Trec dataset, it requires 9 shots to achieve optimal performance. However, on the Trec dataset, regardless of the number of shots used, there is still a certain difference in the performance between our method’s synthetic text and private text. The main reason for this is the highly uneven label distribution in the Trec dataset, with only 86 instances belonging to one label category (Abbreviation), making it challenging to estimate the exact distribution of the private text.

**Synthetic Text Perplexity Check.** In order to enhance the overall readability of the synthetic texts, we also consider whether the parent samples can be decoded successfully as text (in the case they cannot be decoded as text, it might generate gibberish) during iterations. The synthetic text perplexity check operation (Line 18 in Algo. 1) ensures that the current population’s distribution not only approaches the private distribution but also distributes within the successful decoding space. Figure 4 displays final synthetic text under the same settings except for whether the perplexity check operation is performed or not. The more coherent synthetic text demonstrates the significance of this operation.
the synthetic distribution and the private distribution, we use the Wasserstein distance [Santambrogio, 2015] between two embedded distributions. As both the CusText method and the DP-ICL method do not perform text-to-embedding mapping operations, we need to use the same encoder as our approach to obtain their embeddings beforehand. From Figure 3, as the iterations progress, the Wasserstein distance between the synthetic distribution and the private distribution gradually decreases. When reaching the seventh iteration, the distribution of synthetic text obtained by our method is closer to the private distribution compared to the text distributions obtained by all the comparative methods.

![Figure 3: The distance between the synthetic distribution and the private distribution at different iteration. As baseline methods do not involve an iterative process, the distance between distributions is represented by a constant value.](image)

What is the appropriate number of iterations? Although the distance between our synthetic text distribution and the private text distribution decreases as the iterations progress, the maximum number of iterations is constrained by the privacy budget. If we set the number of iterations too high, the limited domain method may output an empty set, preventing the continuation of the iteration. We present the variance information of the voting counts obtained from the neighbor histogram at different iteration in Figure 4, along with the size of the parent set. We can observe that as the variance gradually decreases, the histogram tends to flatten, resulting in fewer parent samples can be selected by the LimitedDomain method.

Can our synthetic text defend against Member Inference Attack? We implement the Member Inference Attack (MIA) from [Duan et al., 2023] on prompts. We study the AGNews dataset and split it in two parts for member and non-member texts. Then, we generate synthetic text sets that closely similar to the private distribution of member text with our DP algorithm. We conduct a 1-shot ICL with one member text or synthetic text on the babbage model. Attacks on both member and non-member texts are repeated 20 times and we represent the probability outputs of correct target classes for member and non-member texts in Figure 5.

We can observe in Figure 5 (a) that when member text is used as a 1-shot demonstration, the predicted probability for non-member text is significantly lower than that for member text. This indicates that using member text in the prompt is susceptible to malicious MIA. However, in Figure 5 (b), when we use synthetic text in the prompt, the predicted probabilities for member and non-member text are relatively close. This suggests that although the distribution of synthetic text is close to that of private text, synthetic text does not leak sensitive information from private text.

5 Conclusion and Future Work

In this work, we propose a novel approach to generate high-readability synthetic text, ensuring differential privacy while maintaining semantic similarity with text in the private dataset. Experimental results demonstrate that using synthetic text as demonstrations for in-context learning incurs only marginal losses in predictive performance compared to using private text. Besides, our synthetic text are also capable of resisting membership inference attacks from malicious users. While it is convenient to invert from embeddings to text, longer text often leads to a higher loss of information within the embeddings, consequently decreasing the quality of synthetic text. In future work, we will explore how to apply our proposed framework to the situation of privacy protection on long text.
Acknowledgments

This work is partially supported by by NSFC (61732006) and Public Computing Cloud, Renmin University of China.

References


[Pan et al., 2020] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language


