Université de Montréal

Local Differentially Private Mechanisms for Text Privacy Protection

par

Fengran Mo

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté en vue de l'obtention du grade de
Maître ès sciences (M.Sc.)
en Informatique

August 31, 2022

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé

Local Differentially Private Mechanisms for Text Privacy Protection

présenté par

Fengran Mo

a été évalué par un jury composé des personnes suivantes :

Esma Aimeur
(président-rapporteur)

Jian-Yun Nie
(directeur de recherche)

Alain Tapp
(membre du jury)

Résumé

Dans les applications de traitement du langage naturel (NLP), la formation d'un modèle efficace nécessite souvent une quantité massive de données. Cependant, les données textuelles dans le monde réel sont dispersées dans différentes institutions ou appareils d'utilisateurs. Leur partage direct avec le fournisseur de services NLP entraîne d'énormes risques pour la confidentialité, car les données textuelles contiennent souvent des informations sensibles, entraînant une fuite potentielle de la confidentialité. Un moyen typique de protéger la confidentialité consiste à privatiser directement le texte brut et à tirer parti de la confidentialité différentielle (DP) pour protéger le texte à un niveau de protection de la confidentialité quantifiable. Par ailleurs, la protection des résultats de calcul intermédiaires via un mécanisme de privatisation de texte aléatoire est une autre solution disponible.

Cependant, les mécanismes existants de privatisation des textes ne permettent pas d'obtenir un bon compromis entre confidentialité et utilité en raison de la difficulté intrinsèque de la protection de la confidentialité des textes. Leurs limitations incluent principalement les aspects suivants: (1) ces mécanismes qui privatisent le texte en appliquant la notion de d_χ-privacy ne sont pas applicables à toutes les métriques de similarité en raison des exigences strictes; (2) ils privatisent chaque jeton (mot) dans le texte de manière égale en fournissant le même ensemble de sorties excessivement grand, ce qui entraîne une surprotection; (3) les méthodes actuelles ne peuvent garantir la confidentialité que pour une seule étape d'entraînement/d'inférence en raison du manque de composition DP et de techniques d'amplification DP.

Le manque du compromis utilité-confidentialité empêche l'adoption des mécanismes actuels de privatisation du texte dans les applications du monde réel. Dans ce mémoire, nous proposons deux méthodes à partir de perspectives différentes pour les étapes d'apprentissage et d'inférence tout en ne requérant aucune confiance de sécurité au serveur. La première approche est un mécanisme de privatisation de texte privé différentiel personnalisé (CusText) qui attribue à chaque jeton d'entrée un ensemble de sortie personnalisé pour fournir une protection de confidentialité adaptative plus avancée au niveau du jeton. Il surmonte également la limitation des métriques de similarité causée par la notion de d_χ-privacy, en adaptant le mécanisme pour satisfaire ϵ-DP. En outre, nous proposons deux nouvelles stratégies de

privatisation de texte pour renforcer l'utilité du texte privatisé sans compromettre la confidentialité. La deuxième approche est un modèle Gaussien privé différentiel local (GauDP) qui réduit considérablement le volume de bruit calibrée sur la base d'un cadre avancé de comptabilité de confidentialité et améliore ainsi la précision du modèle en incorporant plusieurs composants. Le modèle se compose d'une couche LDP, d'algorithmes d'amplification DP de sous-échantillonnage et de sur-échantillonnage pour l'apprentissage et l'inférence, et d'algorithmes de composition DP pour l'étalonnage du bruit. Cette nouvelle solution garantit pour la première fois la confidentialité de l'ensemble des données d'entraînement/d'inférence.

Pour évaluer nos mécanismes de privatisation de texte proposés, nous menons des expériences étendues sur plusieurs ensembles de données de différents types. Les résultats expérimentaux démontrent que nos mécanismes proposés peuvent atteindre un meilleur compromis confidentialité-utilité et une meilleure valeur d'application pratique que les méthodes existantes. En outre, nous menons également une série d'études d'analyse pour explorer les facteurs cruciaux de chaque composant qui pourront fournir plus d'informations sur la protection des textes et généraliser d'autres explorations pour la NLP préservant la confidentialité.

Mots clés: Traitement du langue naturelle, Confidentialité différentielle, Protection de la confidentialité des textes, Méthode de préservation de la vie privée.

Abstract

In Natural Language Processing (NLP) applications, training an effective model often requires a massive amount of data. However, text data in the real world are scattered in different institutions or user devices. Directly sharing them with the NLP service provider brings huge privacy risks, as text data often contains sensitive information, leading to potential privacy leakage. A typical way to protect privacy is to directly privatize raw text and leverage Differential Privacy (DP) to protect the text at a quantifiable privacy protection level. Besides, protecting the intermediate computation results via a randomized text privatization mechanism is another available solution.

However, existing text privatization mechanisms fail to achieve a good privacy-utility trade-off due to the intrinsic difficulty of text privacy protection. The limitations of them mainly include the following aspects: (1) those mechanisms that privatize text by applying d_χ-privacy notion are not applicable for all similarity metrics because of the strict requirements; (2) they privatize each token in the text equally by providing the same and excessively large output set which results in over-protection; (3) current methods can only guarantee privacy for either the training/inference step, but not both, because of the lack of DP composition and DP amplification techniques.

Bad utility-privacy trade-off performance impedes the adoption of current text privatization mechanisms in real-world applications. In this thesis, we propose two methods from different perspectives for both training and inference stages while requiring no server security trust. The first approach is a Customized differentially private Text privatization mechanism (CusText) that assigns each input token a customized output set to provide more advanced adaptive privacy protection at the token-level. It also overcomes the limitation for the similarity metrics caused by d_χ-privacy notion, by turning the mechanism to satisfy ϵ-DP. Furthermore, we provide two new text privatization strategies to boost the utility of privatized text without compromising privacy. The second approach is a Gaussian-based local Differentially Private (GauDP) model that significantly reduces calibrated noise power adding to the intermediate text representations based on an advanced privacy accounting framework and thus improves model accuracy by incorporating several components. The model consists of an LDP-layer, sub-sampling and up-sampling DP amplification algorithms

for training and inference, and DP composition algorithms for noise calibration. This novel solution guarantees privacy for both training and inference data.

To evaluate our proposed text privatization mechanisms, we conduct extensive experiments on several datasets of different types. The experimental results demonstrate that our proposed mechanisms can achieve a better privacy-utility trade-off and better practical application value than the existing methods. In addition, we also carry out a series of analyses to explore the crucial factors for each component which will be able to provide more insights in text protection and generalize further explorations for privacy-preserving NLP.

Keywords: Natural language processing, Differential privacy, Text privacy protection, Privacy-Preserving method.

Contents

List of tables

List of figures

List of Abbreviations

NLP	*Natural Language Processing*
GDPR	*General Data Protection Regulation*
CCPA	*California Consumer Privacy Act*
DP	*Differential Privacy*
SGD	*Stochastic Gradient Descent*
PII	*Personally Identifiable Information*
DP-SGD	*Differentially Private - Stochastic Gradient Descent*
CDP	*Centralized Differential Privacy*
LDP	*Local Differential Privacy*
CusText	*Customized Text Privatization Mechanism*
GauDP	*Gaussian-based Local Differentially Private Model*

FL	*Federated Learning*
RDP	*Rényi Differential Privacy*
GDP	*Gaussian Differential Privacy*
TF-IDF	*Term frequency - Inverse Document frequency*
GloVe	*Global Vectors for Word Representation*
PLMs	*Pre-trained Language Models*
OOV	*Out of Vocabulary*
BERT	*Bidirectional Encoder Representations from Transformers*
LSTM	*Long Short-Term Memory*
Bi-LSTM	*Bi-directional Long Short-Term Memory*
MLP	*Multi-Layer Perceptrons*
CV	*Computer Vision*
USDPA	*Up-Sampling Differential Privacy Amplification*

CLT *Central Limit Theorem*

Acknowledgements

I would like to express my greatest gratitude to my advisor, Professor Jian-Yun Nie, for his allowed autonomy and flexibility in my study during the global pandemic period. He is a super nice advisor and can always give me useful advice when I have problems. He teaches me how to think and probe the mystery of research and lead me to the door of science. His insightful guidance supports me in exploring an interesting research topic that is totally new for us. I learned and benefited a lot from him.

At the same time, I would like to thank my family, for their unconditional support, careful listening, and encouragement. They brought me indispensable happiness. They are the beliefs of my study career and school life.

Chapter 1

Introduction

1.1. Data Privacy Concerns

Natural Language Processing (NLP) based on neural models has given rise to a new generation of approaches to deal with problems such as sentiment analysis, question answering, semantic matching, and so on. These applications may require a massive amount of personal data during the training stage, as well as personal queries sent to the service providers during the inference stage. The input text data for these applications often contain sensitive information [**36**], which raises the privacy risks such as potential personal privacy leakage and data abuse during data sharing and collecting. In general, better services can be provided if the server can collect more new data for model renewal.

However, the privacy-conscious people might not agree to release their data which might contain personal information to service providers without privacy guarantees. Meanwhile, many data protection initiatives and privacy laws have been launched in recent years, such as the General Data Protection Regulation (GDPR) [**64**] and the California Consumer Privacy Act (CCPA) [**1**]. This imposes obligations to the service providers of NLP applications and makes their data collection more difficult unless they can address the privacy concern of the data owners. Thus, the personal sensitive information should have a guarantee not be shared with unwarranted parties or be attacked by the potential eavesdroppers.

In practice, data owners are allowed to privatize their texts locally by a certain text privatization mechanism before sending them to service providers. Then, the service providers can process the privatized texts or privatized representations for further use. For example, to design a privacy-preserving search engine, the query is first processed by a privacy protection mechanism. Then the query (or query embedding) is sent to a search engine. The privacy-preserving search engine will work with the privatized query and find relevant documents to return to the user. What is required is to send queries so that the search engine cannot read the private information from the queries, and the returned results are the most similar to that with the unprivatized query. The procedure overview of different practical application

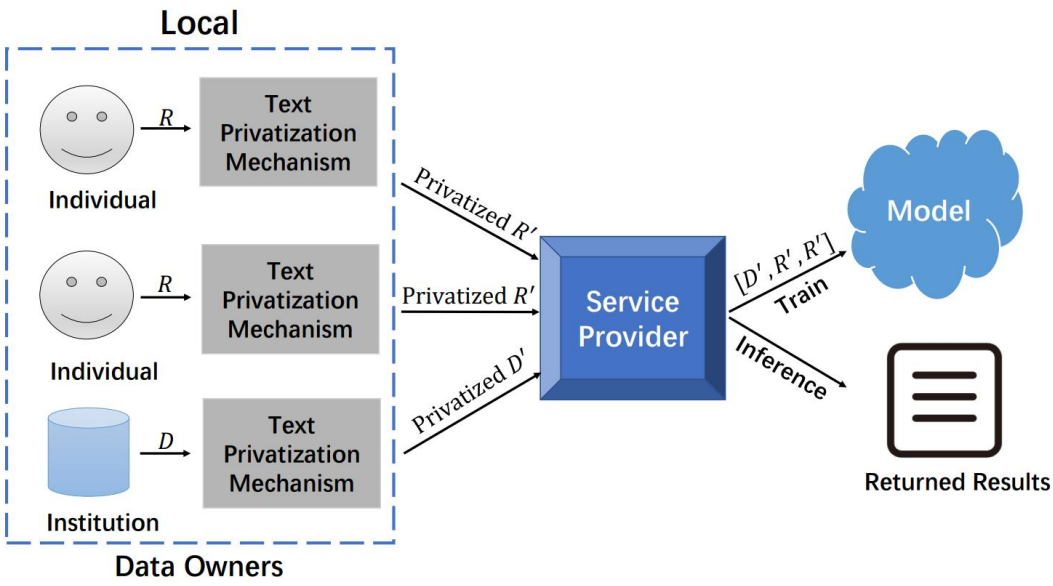


Fig. 1.1. An overview of local privacy scenario, in which a raw text is privatized before being sent to the service provider for further use.

scenarios can be summarized as illustrated in Fig. 1.1. The individual record R and the data collection D should be privatized into R' and D' by a text privatization mechanism before releasing. Then the service provider can leverage them for various goals. Many studies, however, have discovered privacy violations in neural models no matter whether raw input texts or text representations are sent to the service providers [**67, 15**]. Previously, it has been shown that simple anonymization techniques, such as the removal of personal sensitive information or protected attributes, fail to preserve data privacy [**70, 40, 46, 10, 39**]. Instead of simply anonymizing raw data, the utilization of the learned representation as abstract real-number vectors encoded by the neural models does not provide a guarantee of safety either. The advanced attackers have been shown to be able to recover private information from the deep neural representations [**42, 18, 16, 68, 57**]. Therefore, the aforementioned data privacy concerns necessitate further research on exploring privacy-preserving NLP methods with provable and quantifiable privacy guarantees for text protection [**31, 69**].

1.2. Outlook of Our Methods

Currently, many differentially private text privatization mechanisms [**30, 62, 81, 35, 52, 44**] have been proposed to address the privacy concern of the data owners. The privatization mechanism aims to protect the private information in the text before releasing it for further use and the privacy of the original input text in those mechanisms is guaranteed by differential privacy (DP) [**24**], which becomes a de facto standard for privacy protection. The DP randomizes the computation process to stabilize the output in the face of changes to input data, ensuring that the adversary can hardly tell if an individual data item (e.g. a token or

a sequence) is in the dataset or not by looking at the computation output (e.g. privatized text or latent representation). Recently, the DP has been integrated into the deep learning training stage as the *Differentially Private-Stochastic Gradient Descent* (DP-SGD) [**5**] algorithm proposed. It randomizes back-propagation with calibrated noise to limit what could be breached from the training data when revealing the model. As a result, the model parameters can be viewed as a sanitized release, with individual training data obscured but the model still remains functional. However, due to the calibrated noise required for DP, it has been recognized that DP mechanisms invariably significantly reduce the downstream task performance, raising the privacy-utility trade-off issue [**23**]. Instead, the DP-SGD is based on a centralized differential privacy (CDP) setting which assumes a trusted service provider can directly collect and process customers' text data. The concurrent works [**41, 22, 79, 7**] based on the CDP setting are also inapplicable for our considered scenarios where the users have to privatize their texts locally before sending them to the untrusted server providers. Instead, normally the trusted third party is not available so users should use the local privatization approach. Therefore, our methods are designed based on local differential privacy (LDP) [**21**] settings which have been deployed in many real-world applications, such as Apple IOS [**2**], Uber [**4**] and Google Keyboard [**3**].

Currently, two research lines are conducted based on LDP settings. On one hand, to reduce the impact of DP mechanism on the original semantics and syntax as much as possible, some previous researches [**44, 52, 35**] focus on producing differentially private text representations. The rationale behind this is to generate random DP noise and add it to the raw text representation. Then, the text representation in semantic space is changed and its original position (representation vector) cannot be known. However, they only consider either training or inference phrases whose calibrated noise for DP protection is based on training or inference datasets. For example, the size of the training and inference datasets are different, and the noise calibrated for one might not applicable for another one, i.e. the noise required to protect one query and one hundred queries should be different. Besides, the data usage of the two phrases is also different as all queries need to be processed in the inference stage while we might not need to iterate all data on training, which could make the construction of the mini-batch different. Thus, no existing approaches can cope with both the training and inference phases. In addition, these existing works lack some specific DP techniques (e.g. DP amplification) to enhance the privacy protection level and improve the model utility. Another line of research privatizes the text data from the original. The basic idea of achieving privacy protection is to generate privatized text by replacing the original tokens in the text sequentially with new tokens (can be a character, a subword, a word, or an n-gram) that are sampled from output token sets. The privacy and the utility of these existing methods [**30, 62, 81**] are guaranteed by d_χ-privacy [**17**] which is a relaxation of the original DP definition. The d_χ-privacy inherits the main idea of DP to protect the original

token from being inferred. It further improves the utility of privatized text by giving higher sampling probability to tokens that are semantically closer to the original one, so as to preserve more information from the input text. However, the d_χ-privacy has strict requirements for similarity metrics and privatizes each token in the text equally by providing the same and excessively large output set.

Currently, most existing methods cannot achieve a good privacy-utility trade-off, i.e., either large privacy cost with insufficient protection or small privacy cost with unsatisfiable model accuracy. To address the aforementioned problems, we design our local differentially private methods for text privacy protection via suitable DP mechanisms following both these two research lines: (1) directly privatize the user's original input text by token sampling; (2) protect the computation results of user's text by producing differentially private latent representation.

1.2.1. Differentially Private Text

To directly privatize the user's original input text via token sampling locally, we propose a new Customized Text privatization mechanism named *CusText* to convert the raw user data into differentially private text. The whole procedure is shown and discussed in Sec. 3.2. The CusText assigns each input token with a unique output set for customization to provide adaptive privacy protection at token-level. This is the main difference compared with existing approaches where they use the whole vocabulary as the output set. Assigning unique output sets for each input token can avoid the over-protective problem and boost the downstream task utility. CusText also overcomes the limitation of the applicability of similarity metrics caused by d_χ-privacy notion, by turning the mechanism to satisfy ϵ-DP based on a carefully designed score function. Furthermore, two new text privatization strategies are provided to boost the utility of privatized text on downstream tasks without compromising privacy.

1.2.2. Differentially Private Latent Representation

To locally protect the computation results of the user's text by producing differentially private latent representation, we further propose a Gaussian-based Local Differentially Private model for NLP named *GauDP*. We consider a similar scenario to the previous works [**18, 44**] and extend it into a complete LDP-NLP tasks pipeline as shown and discussed in Sec. 4.2. Among our GauDP model, a novel LDP layer is deployed on the user side to randomize the intermediate output, i.e. latent representation, for training and inference's forward computations, respectively. The DP composition and DP amplification techniques should be carefully designed when producing differentially private latent representation. Thus, we propose novel sequence-level sub-sampling and up-sampling DP amplification techniques based on the Gaussian mechanism with μ-Gaussian DP framework

for both the training and inference stages, that reduce the privacy cost parameter ϵ to less than 10 across the entire training/inference dataset. By contrast, the same data privacy cost parameter ϵ can only be guaranteed in either the training or inference one step in the previous literature [**44, 62, 30**] due to the lack of DP techniques. In other words, thousands of additional training or inference steps can be carried out in our proposed method with the same privacy level as existing methods.

1.3. Organization

The organization of the thesis is laid out as follows:

In Chapter 1, we introduced the data privacy concerns in the deep learning era and the necessity of text privacy protection, then outline our methods.

In Chapter 2, we first illustrate some typical privacy risks in NLP and some popular privacy-preserving methods. Then we introduce the necessary background knowledge of differential privacy as well as its properties to solve privacy problems. Finally, we review previous works on differentially private NLP and corresponding settings related to our works.

In Chapter 3, we first introduce how to achieve privacy protection by directly privatizing text via token-to-token replacement in our customized text privatization mechanism CusText. Then, we describe our experimental settings and present the privacy-utility results as well as the comparisons.

In Chapter 4, we first depict the LDP-NLP task pipeline and our designed Gaussian-based algorithms for noise calibration, which achieves sentence-level protection for differential private latent representation. Then, we present the experiment results as well as the comparison with other methods.

In Chapter 5, we conclude the thesis and give some possible improvement directions for future work.

Chapter 2

Related Work

In this chapter, we first introduce the potential privacy risks in NLP to illustrate the defense goals of privacy-preserving methods. Then, we introduce the current privacy-preserving methods as well as the important background knowledge of differential privacy (DP) and how these techniques can solve privacy problems, which is the foundation of our text privatization mechanisms. Finally, we review the differential private NLP research progress, which is closely related to our proposed models and corresponding settings.

2.1. Privacy Risks in NLP

The exposure of potential privacy risks raises public concern. In this section, we will first describe three attacker goals which are also the defense targets of our LDP-based methods.

2.1.1. Local Privacy Risks

One of the most obvious risks is that the complete data from the data owners are stored in a server in a raw form without any perturbation or transformation, which means that the user's private data is completely unguarded and exposed to various possible attacks. This should be avoided as much as possible in real situations. Besides, when the data owners do not trust any third parties including the server providers, which is referred to as a local privacy setting [**21**], raising *local privacy risks*. Private information might leakage after publishing personal data. Thus, a privacy-preserving mechanism should be designed for the system to enable the data owners to privatize their personal data before sharing or being collected for processing to defend against local privacy risks.

2.1.2. Reconstruction Attacks

Reconstruction attacks aim to eavesdrop on the raw data in the communication process or recover the raw data according to its encoded features (e.g. latent representation) which can be linked to the original form. After the attackers obtain the raw data, personal privacy

is leaked and this can further cheat the machine learning models by reconstructing fake data. For example, Jia et al. [**37**] are the first to consider the adversarial attacks on deep neural networks for text-related tasks, which modifies the original input data to evaluate the system robustness. Besides, the original form can also be inverse by reconstruction attacking. Carlini et al. [**16**] reconstruct verbatim texts of training data through a powerful black-box attack on GPT-2 [**63**] and Song et al. [**68**] also recover sensitive attributes or partial raw text from the output of a language model without any prior knowledge of the input text patterns. Therefore, data features in plaintext cannot be considered safe and these studies suggest that we should avoid explicitly storing feature vectors or any other data forms that may reveal original data information.

2.1.3. Membership Inference Attacks

In addition to obtaining the original data or its corresponding features, an attacker may also want to infer whether a certain data sample was used to build the machine learning model, which is referred to as membership inference attacks [**67, 78, 65**]. This attack exploits differences in model predictions due to the inclusion or exclusion of a sample in the dataset. An attacker can use membership inference attacks to know whether an individual's records were used to train the model according to the computation results as well as the gradient change of different input data. Zhao et al. [**82**] evaluates various differential privacy implementations against membership inference attacks and measures their ability for defending, which provides some clues for implementing privatized position among the whole model pipeline.

2.2. Privacy-Preserving Methods

Different privacy-preserving methods can be adapted for specific attack scenarios. In this section, we introduce the general idea of four main privacy-preserving methods and mainly focus on illustrating the suitability of **Differential Privacy** as well as its basic definition under the local privacy protection scenarios.

2.2.1. Differential Privacy

Differential privacy (DP) [**24**] has recently been considered a promising strategy for privacy-preserving in machine learning because it has a rigorous theoretical model and can provide a provable privacy guarantee for individuals, which benefits from the most solid theoretical basis compared with other privacy-preserving methods [**70, 77, 32**]. It has become the de facto standard of privacy definition. Besides, DP achieves privacy-preserving by adding a calibrated amount of noise to the model or output results according to the concrete mechanisms instead of simply anonymizing the individual data. From a computational point

of view, DP can solve the disadvantage of the excessive overhead of cryptography [**27**]. It also solves the privacy leakage of the data calculation results rather than the calculation process in cryptography. DP aims to prevent membership inference and data reconstruction attacks and can be adopted in both centralized and local privacy settings. For distributed computing machine learning algorithms, DP can protect the original data of multiple input parties [**25**]. Considering the advantages of DP, it is the most suitable method for our privacy requirements and thus our text privatization mechanisms for NLP are guaranteed by DP in this thesis.

Definition of Differential Privacy. In the field of NLP, a random DP algorithm's output is stabilized to the point where the presence or absence of any specific data item, such as a token or a sequence is hardly distinguishable. The type of data item is the DP granularity. The greatest possible divergence between two output distributions of DP algorithms when applied to two data inputs that differ by arbitrary data item of the DP granularity describes the DP protection level. An intuitive description is given in Fig. 2.1. By injecting random noise via sampling from Laplace distribution, the distribution of two input data samples would become similar which makes the attackers hard to distinguish them. A concrete instance of DP intuition for NLP is also shown below. Assuming two sentences have a one-word difference, after applying the DP mechanism to each token, the attackers might not infer whether "marry" or "divorce" are included in the sentence, because of the randomization of the two distributions.

Text One: when did spielberg and irving **marry**?

Text Two: when did spielberg and irving **divorce**?

More precisely, differential privacy is defined as follows:

Definition 1. *(Differential Privacy* [**24**]*) Let $\mathcal{X}$ and $\mathcal{X}'$ be adjacent data inputs that differ in one data item, then the randomized algorithm $\mathcal{M}$ satisfy (ϵ, δ)-DP if for possible arbitrary subset $\mathcal{Y}$ of the all possible output of $\mathcal{M}$:*

$$\Pr[\mathcal{M}(\mathcal{X}) \in \mathcal{Y}] \leq \mathrm{e}^{\epsilon} \Pr[\mathcal{M}(\mathcal{X}') \in \mathcal{Y}] + \delta. \tag{2.2.1}$$

where the probability $\Pr[\cdot]$ is taken over the randomness of $\mathcal{M}$, and $\epsilon \geq 0$.

In the above definition, the privacy protection level is characterized by calculating a pair of DP privacy parameters[1] (ϵ, δ) in differential privacy, which defines an upper bound on privacy leakage because it satisfies the above conditions and further defines the worst-case protection level for a single data sample. Intuitively, this means that we cannot easily tell whether the result of the random function $\mathcal{M}$ comes from $\mathcal{X}$ or $\mathcal{X}'$ by a random DP

[1]Also denote the DP privacy cost, DP privacy level, and DP budget in literature.

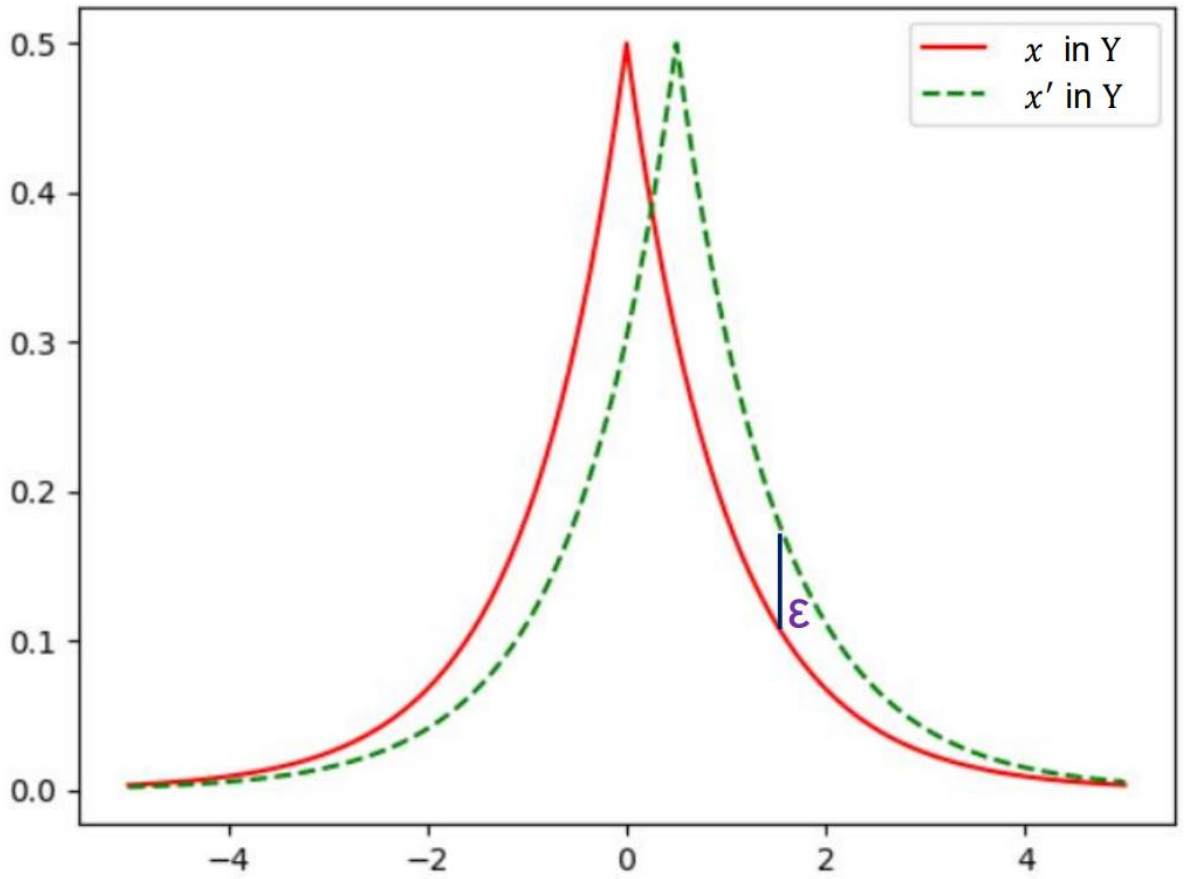


Fig. 2.1. An intuitive description of protecting private data from distinguishing two data distributions. The greatest possible divergence between two output distributions indicates privacy level ϵ.

algorithm's output. Therefore, it is almost impossible for an adversary to infer the existence of any particular data sample in the input dataset.

Specifically, privacy parameter $\epsilon \geq 0$ depicts the upper limit of the difference between the two output results obtained by the random function $\mathcal{M}$ acting on two data inputs $\mathcal{X}$ and $\mathcal{X}'$. The smaller the ϵ, the smaller the difference, the higher privacy protection level, but the lower the usability of the results as a trade-off. Without considering δ, when $\epsilon = 0$, two data inputs after DP processing will output the same probability distribution. On this basis, the attacker will be completely unable to distinguish the input data, but the noise that needs to be added tends to be close to ∞. In this case, such noise will overwhelm the data, making the impact of the data to be null, and making the model learning meaningless. When $\epsilon \to \infty$, the noise that needs to be added is infinitely close to 0. In this case, the attacker can easily distinguish two input datasets and then deduce the relevant information of the original data, which does not achieve the effect of privacy protection. When $\delta = 0$, (ϵ,δ)-DP becomes ϵ-DP, which is also the strictest definition of DP at the beginning. The value δ can be interpreted

as the likelihood of not achieving DP and allows for suppressing the long-tail effect of the distribution. In other words, there is a certain probability of failure for long-tailed samples but we reduce the noise that needs to be added by relaxing the restriction. Usually, we set δ to be less than or equal to the reciprocal of the number of samples to ensure that the differential privacy mechanism fails as a small probability event. Specifically, because of the mathematical constraints and proof within the Laplace mechanism [**24**] and Gaussian mechanism [**25**] (refer in Sec. 2.3.1), we should apply the former in conjunction with ϵ-DP and the latter for (ϵ,δ)-DP.

Besides, the $d_{\mathcal{X}}$-privacy is a relaxation of DP definition which is widely used in the text field [**62, 30, 81**] as it can intuitively measure the distance between two input tokens. The $d_{\mathcal{X}}$-privacy allows the indistinguishability of the output distributions to be scaled by the distance between the respective inputs. Formally,

Definition 2. *($d_{\mathcal{X}}$-privacy)* [**17**] *Let x,$x' \in \mathcal{X}$ be adjacent data inputs that differ in one data item, given a privacy parameter ϵ, and a distance metric d, the randomized algorithm $\mathcal{M}$ gives $d_{\mathcal{X}}$-privacy if for a possible arbitrary subset $\mathcal{Y}$ of all possible output of $\mathcal{M}$*

$$\Pr[\mathcal{M}(x) \in \mathcal{Y}] \leq \mathrm{e}^{\varepsilon d(x,x')} \Pr[\mathcal{M}(x') \in \mathcal{Y}] \tag{2.2.2}$$

So far, we have introduced the standard definition of DP as well as its relaxation. Then, we introduce how to exploit general DP with deep learning methods.

Differential Privacy Deep Learning. After Abadi et al. [**5**] proposed the Differentially Private-Stochastic Gradient Descent (DP-SGD), DP can be used for neural network model training within an acceptable accuracy loss. The general neural model parameters θ training is optimized through Stochastic Gradient Descent (SGD). For each training step t, the calculation result of the gradient for data sample $\mathcal{X}_t$ sampled from dataset $\mathcal{X}$ is denoted as $g_x = \frac{\partial f_x}{\partial \theta}$, where f_x is the loss function of the specific task. Since the gradient generated by back-propagation has no fixed range of variation, in order to calibrate the noise power required by the DP mechanism, the paradigm proposed by the DP-SGD algorithm first clips the ℓ_2 norm of the gradient. That is to limit the maximum value of the gradient norm by proportional reduction. The clip function $CL(.)$ is defined as

$$\widetilde{g}_x = \mathrm{CL}\left(g_x; C\right) = g_x \ / \max\left(1, \frac{C}{\|g_x\|_2}\right), \tag{2.2.3}$$

The DP-SGD algorithm makes the maximum ℓ_2 norm of the gradient generated by back-propagation to become C through the clip operation, then the sensitivity Δ (refer to Sec. 2.3.1 for details) of the back-propagation function is limited to C. After clipping, the noise is calibrated through the DP mechanism as $\xi_t \sim \mathcal{N}(0, \sigma^2 I_d)$, where σ^2 is the variance of noise power and d is the dimension of the gradient. Then, the parameter θ updating of DP-SGD

algorithm [**5**] at step t is formulated as

$$\theta_t = \theta_{t-1} - \eta \frac{1}{|\mathcal{X}_t|} \left(\sum_{x \in \mathcal{X}_t} \widetilde{g}_x + \xi_t \right), \quad t \in [1, T]. \tag{2.2.4}$$

According to the DP post-processing property (refer to Definition 7), preserving the gradient provides the same level of privacy protection for the output model. Therefore, DP-SGD has become a standard paradigm for training differentially private neural networks. However, since DP-SGD training is inefficient and cannot provide local privacy protection in some cases, we only compare the effectiveness with it rather than applying it in our work.

2.2.2. Other Privacy-Preserving Methods

2.2.2.1. Anonymization. Anonymization [**70, 40, 46**] is a data processing technique that removes or modifies personally identifiable information (PII). The PII is generally understood as any information that can directly identify an individual, and such information has different degrees of importance depending on the degree of identifiability and sensitivity. For example, information such as names and email addresses are highly identifiable but low-sensitivity, and posting such information usually does not harm individuals. In the contrast, information such as location data and personal health records are low-identifiable but high-sensitivity and need to be treated with caution. The identifiability and sensitivity of PII also depend on the composite effect of the text background and data mixing. For example, posting someone's name from a Facebook fan database might have low risk, but posting someone's name on a list of political dissidents carries significantly more risk. When multiple pieces of data are combined, the value of the information will also change. For example, if you look at a database of purchase records alone, it is difficult to connect to any specific individual, but combining location information or credit card numbers will greatly increase the recognizability and sensitivity.

The anonymization which is also referred to as de-identification is to prevent re-identification, in other words, to anonymize the data so that the data cannot be used to identify any individual. Typical anonymization methods include data redaction and statistical noise. The former can directly delete all personal or sensitive data or perform pseudonymization for the PII by replacing identifiable data with random or algorithmically generated pseudonyms. For example, in the search engine log, the system tends to substitute the true user name as a unique anonymized ID. Compare with direct delegation, pseudonymization can somehow keep the availability of data. The statistical noise refers to a certain number of individuals with an indirect identifier, and the best practice is to use the same unique identifier for no less than ten entries, thus making re-identification difficult. The most common technique to add statistical noise to a dataset is *generalization*, such as substituting continents for country names and numerical ranges for exact values.

However, these method has been proved in some literature [**58, 71**] to be unable to provide sufficient privacy guarantee especially when the adversaries own auxiliary side information. For instance, even if the same pseudonym is reused throughout the dataset, the effectiveness of pseudonymization is reduced because each occurrence of the pseudonym increases the chances of finding relationships between variables. In other cases, the algorithm used to generate the pseudonym has the opportunity to be cracked by a third party, or the algorithm itself has loopholes. Besides, because of the theoretical and empirical limitations [**6, 13**], anonymization does not apply to high dimensional data, and thus might not be suitable for neural models compared with DP.

2.2.2.2. Federated Learning. Federated learning (FL) [**77**] is designed to carry out efficient machine learning among multiple participants or multiple computing nodes under the premise of ensuring information security. It is commonly used during big data sharing to protect the privacy of terminal data and personal data and ensure legal compliance. Generally, the distributed optimization process can be described as

$$\min_{w} \left\{ \mathcal{L}(w) = \sum_{k=1}^{N} p_k \mathcal{L}_k(w) \right\}$$

where N is the total number of user devices and p_k is the weight of the k^{th} device. Assuming that the data on the k^{th} device is $X_k = (x_{k,1}, x_{k,2} \cdots x_{k,n_k})$, then the optimization function for local training can be

$$\mathcal{L}_k(w) = \frac{1}{n_k} \sum_{j=1}^{n_k} \ell(w; x_{k,j})$$

Since the data comes from various devices, the data distribution no longer satisfies the traditional machine learning assumption that the data is independently and equally divided, so the occurrence of Non-IID is very common. Non-IID has profoundly affected the final training results of federated learning. Different data distributions can lead to significant skewness between devices or locations. Data augmentation, regularization on the server, and scheduled client participation during training can be applied to alleviate such issues.

Centralized FL is used by most NLP applications such as keyboard word prediction [**48, 43**]. The central server is used to coordinate the different steps of the algorithm and coordinate all participating clients/devices during the learning process. The server is responsible for selecting clients/devices at the beginning of the training process and aggregating the received model updates. The whole procedure can be depicted as Fig. 2.2, where each local participant train a model for a specific purpose Q based on their own data D_i and service, and communicate with the global service with gradient R_i and updated Q to produce a global model. On decentralized FL, clients/devices can coordinate with each other to obtain a global model. This setup prevents a single point of failure as model updates

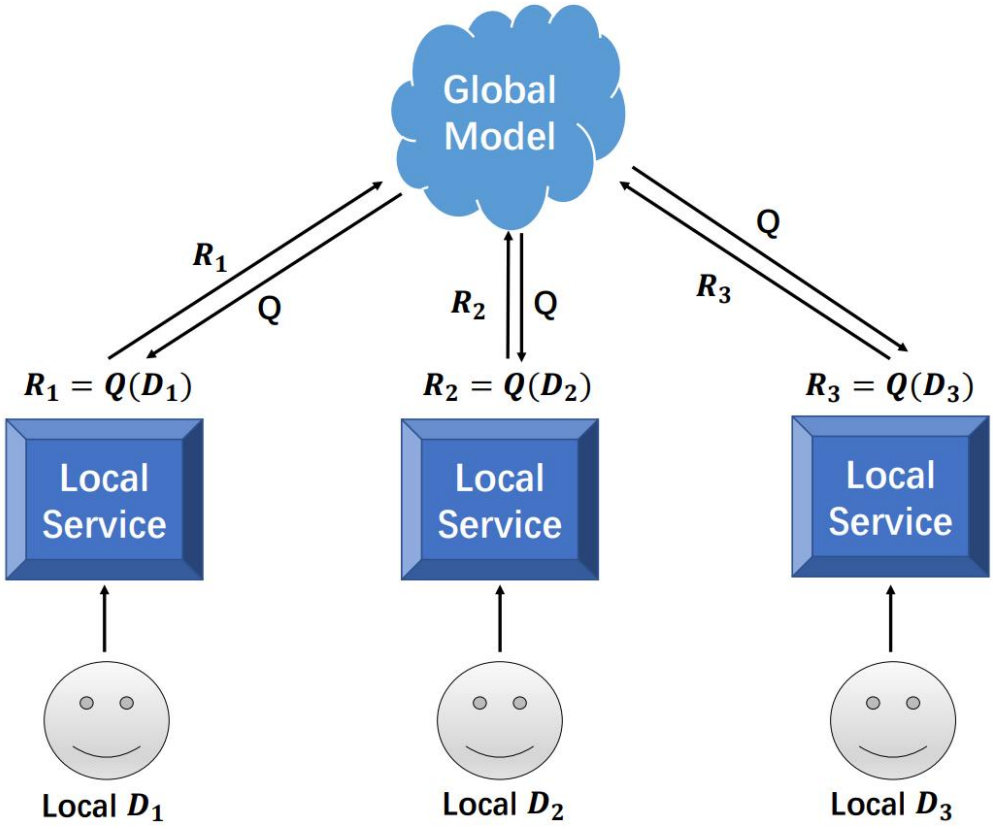


Fig. 2.2. An overview of centralized FL training framework.

are only exchanged between interconnected nodes without the need for orchestration on a central server. The FL training framework ensures the user data is left on users' local devices without sacrificing local data privacy [**47**], which can also satisfy local privacy settings. However, FL still has its privacy flaws. Though the transmission of gradients replaces the sharing of original data, the gradients might still leak available information and be attacked to recover the original data [**45**]. Therefore, to ensure the required privacy settings, previous works [**48, 12, 9**] leverage the DP mechanisms to randomize the data in gradient form when applying FL. In our works, we consider a local-global situation similar to centralized FL.

2.2.2.3. Adversary Training. Adversarial training is originally proposed by Goodfellow et al. [**32**] as a way to defend against adversarial attacks. Since the private information can take the form of keywords explicitly contained in the text or be implicitly included in the latent representation. For example, demographic information about the author of a text can be predicted with above-chance accuracy from linguistic cues in the text [**61**]. Because an attacker can access to the hidden representations which are non-intentional and incidental learned by a network, they may exploit the latent representation to recover information about the input. For privacy-preserving purposes, some existing works [**18, 42**] train the models to learn private text representation to enable them not to memorize the unintended information

via adversarial training. The goal is to prevent an attacker from recovering information about the input text as reconstruction attacks. Coavoux et al. [**18**] provides an example of a potential application that would be a spam detection scenario where the service provider does not access verbatim emails sent to users, only their vector representations. These vector representations should not be used to gather information about the user's contacts or correspondents, i.e. protect the user from profiling. However, this principle provides only empirical improvements in privacy, without privacy guarantees in mathematical form and theoretical proof as DP, which might not be enough to convince users.

2.3. Differential Privacy Accounting

The core of DP is its accounting framework. The accounting is used to assure accurate measurement of privacy protection level during the whole DP training and inference procedure. Specifically, it ensures the conversion between applying several DP mechanisms and the total privacy protection level[2] ϵ. Generally, the DP accounting framework consists of three parts. The first part is how to describe the relation between the required noise amount of a single DP mechanism and the DP protection parameter ϵ. The second part is how to realize the conversion between the composed results of multiple DP mechanisms and DP protection parameter ϵ. The third part is how to combine randomized mechanisms (such as sampling in our works) to achieve DP amplification, a practical technique to reduce the required noise and then promote the downstream task performance, though it is not mandatory. Then we can describe the privacy cost more accurately and achieve better utility.

In the following, we will first introduce the different DP mechanisms and several desirable DP properties including DP composition, DP amplification, and DP post-processing. They are used for calibrating accurate noise to ensure the final model has accurate and strong privacy protection while retaining acceptable model utility. We will also introduce two widely used advanced privacy accounting methods which inherit the original DP properties and can produce accurate and quantifiable DP guarantees. We design our methods and calibrate the privacy protection level base on these DP accounting frameworks in Chapter 4.

2.3.1. DP Mechanisms

The DP mechanisms are used for generating different types of random noise to achieve DP protection. Before introducing DP mechanisms, we first introduce the concept of sensitivity.
Definition 3. *(Sensitivity) Given a deterministic vector-valued computation function f and two arbitrary data inputs x and x', the sensitivity Δ of f, which is the greatest variation*

[2]If using the Gaussian mechanism, it becomes (ϵ,δ)

output for only one data item change in the worst case is given by

$$\Delta = \max_{x,x'} ||f(x) - f(x')||_p. \tag{2.3.1}$$

where $||f(x) - f(x')||_p$ refers to the ℓ_p-norm between $f(x)$ and $f(x')$.

The sensitivity plays an important role in the noise calibration process because the quantitative noise generated by DP mechanisms can only be made after estimating the sensitivity of the protection target. As the protection level ϵ is based on the distribution divergence of two protection targets according to the DP definition, an object with larger sensitivity Δ needs to be added more noise to achieve the same protection level. This is because the greatest variation output for only one data item change will become larger. Therefore, estimating the sensitivity Δ correctly and tightly for input x is crucial for privacy guarantees and it will affect the required amount of noise. In terms of NLP, the x should be a token or sequence depending on the type of task and the $f(x)$ is the embedding of the input. Thus, the sensitivity Δ estimation is based on a representation unit.

After introducing the concept of sensitivity, we now introduce the three common DP mechanisms: the Laplace mechanism, the Gaussian mechanism, and the Exponential mechanism. The first two mechanisms are widely used for numerical results while the last is used for discrete results. Their utilization will affect the final DP guarantee.

Definition 4. *(Laplace Mechanism)* [**24**] *Given a computation function $f(x) := x \to \mathbb{R}^d$, a randomized algorithm $\mathcal{M}$ with Laplace mechanism can be defined as*

$$\mathcal{M}(x) = f(x) + Lap(0, \frac{\Delta}{\epsilon})$$

where the noise $Lap(0, \frac{\Delta}{\epsilon})$ can be viewed as drawn from the Laplace distribution with the center of 0 and the scaling of $\frac{\Delta}{\epsilon}$. The sensitivity Δ of the Laplace mechanism is l_1-norm. Then $\mathcal{M}$ can provide ϵ-DP.

Definition 5. *(Gaussian Mechanism)* [**25**] *Given a computation function $f(x) := x \to \mathbb{R}^d$, a randomized algorithm $\mathcal{M}$ with Gaussian mechanism can be defined as*

$$\mathcal{M}(x) = f(x) + \mathcal{N}(0, \sigma^2)$$

where the noise variable is drawn from the Gaussian distribution with the standard deviation of $\sigma = \frac{\Delta\sqrt{(2ln(2/\delta))}}{\epsilon}$. The sensitivity Δ of the Gaussian mechanism is l_2-norm. Then $\mathcal{M}$ can provide (ϵ,δ)-DP.

Definition 6. *(Exponential Mechanism)* [**49**] *Given a score function $u(\cdot,\cdot) : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, an Exponential mechanism $\mathcal{M}(\mathcal{X},u,\mathcal{Y}) : \mathcal{X} \to \mathcal{Y}$ satisfies ϵ-DP if it samples an output $y \in \mathcal{Y}$ to perturb the input $x \in \mathcal{X}$ with probability proportional to*

$$e^{\frac{\epsilon u(x,y)}{2\Delta u}}, \tag{2.3.2}$$

where $u(x,y)$ denotes the score for each input and output data pair (x,y) and $\Delta u := \max_{y \in \mathcal{Y}} \max_{x,x' \in \mathcal{X}} |u(x,y) - u(x',y)|$ denotes the sensitivity within Exponential mechanism.

To sum up, Laplace and Gaussian DP mechanisms are used to generate random noise by sampling from a specific distribution. Then the noise is injected into the protected object (e.g. text representation) for achieving the DP guarantee. For those two, the Laplace mechanism is popular for most previous text protection works [**44, 62, 30**] because its DP accounting is easier to implement. However, we try to explore the superior properties of the Gaussian DP mechanism for our methods. The Exponential mechanism is used to generate a probability distribution for all tokens in the output set (usually the whole vocabulary). Then the text privatization is performed via sampling the token based on the probability to substitute the sensitive tokens. The main problem is that the existing works still cannot achieve a good privacy-utility trade-off. To design more advanced DP protection algorithms for text with suitable DP mechanisms, in this thesis, we use the Exponential mechanism for CusText and the Gaussian mechanism for GauDP, which are introduced in Chapter 3 and Chapter 4.

2.3.2. DP Composition

The DP composition is used to calculate the total DP privacy parameters after multiple DP mechanisms are successively applied to the protection target. It plays an important role in designing DP algorithms. It can be used to control and quantitatively analyze the DP privacy cost required in use. To complete an NLP downstream task, both the training and inference stages must perform a series of computation steps on the private train/test dataset in a neural model, with each computation step potentially based on the results of previous computation steps on the same dataset. When producing differential private latent representation, even if each step i is DP protected with privacy cost (ϵ_i, δ_i), providing all step's outputs together linearly boosts the total privacy cost for the whole training/inference procedure. The total privacy cost $(\sum_i \epsilon, \sum_i \delta)$ is equivalent to the sum of each step's privacy cost according to the original DP composition Theorem 1.

Theorem 1. *(DP Composition) When any k DP mechanisms $\mathcal{M}_i$ satisfying $\epsilon_1 - DP$, ..., $\epsilon_k - DP$ are applied to the same dataset, the whole DP algorithm satisfies $(\sum_{i=1}^{k} \epsilon_i)$-DP.*

In other words, the computation of privacy degradation as the number of steps increases is DP composition [**50**]. Thus, a high privacy cost may no longer guarantee privacy as the adversary can easily distinguish the distribution divergence of the input data. However, the training stage in a neural network involves intrinsic repetitive mini-batch operations for an algorithm. Each mini-batch should be applied with a DP mechanism. Therefore, we need to precisely account for the appropriate privacy cost cap for every training stage through DP composition, and a tight composition methodology is needed for a better privacy-utility trade-off. In the following, we introduce general DP composition methods.

Basic Composition [**25**]. Suppose $\mathcal{M} = (\mathcal{M}_1,\ldots,\mathcal{M}_T)$ is a sequence of DP mechanisms, where $\mathcal{M}_i$ satisfies ϵ-DP, then the DP mechanism $\mathcal{M}$ satisfies $(T\cdot\epsilon)$-DP. The basic composition is the most straightforward way to compose several DP mechanisms. It provides a way to generate the final privacy guarantee linearly but might degrade utility in the meanwhile.
Advanced Composition [**25**]. Suppose $\mathcal{M} = (\mathcal{M}_1,\ldots,\mathcal{M}_T)$ is a sequence of DP mechanisms, where $\mathcal{M}_i$ satisfies (ϵ,δ)-DP, for all $\epsilon,\delta,\delta' \geq 0$, then the DP mechanism $\mathcal{M}$ satisfies $(\epsilon', T\delta+\delta')$-DP under T-fold advanced composition with

$$\epsilon' = \sqrt{2k\ln(1/\delta')}\cdot\epsilon + k\cdot\epsilon\left(e^{\epsilon}-1\right). \tag{2.3.3}$$

where the T, k, δ and δ' are hyper-parameters.

The advanced composition leverage additional hyper-parameters k, δ, and δ' to compose different DP mechanism in a non-linear way. Compared with basic composition, it can provide a stronger privacy guarantee under the same condition. Since the basic composition and the advanced composition are not the most superior DP composition so far, we only give the basic introduction here. More detailed proof can be found in the literature [**25**].
Other Composition. In addition to the above two common DP composition methods, there are also moments accountant [**5**] proposed together with DP-SGD, the Rényi DP [**53**] based on Rényi divergence, and the μ-Gaussian DP [**20**] based on the central limit theorem approximation as well as hypothesis testing. Since the DP-SGD is not applicable for the local privacy setting as previously mentioned, we mainly focus on leveraging the μ-Gaussian DP approach in Chapter 4 and compare it with the Rényi DP. The details of these two composition procedures are described in the later Sec. 2.3.5 and Sec. 2.3.6.

2.3.3. DP Amplification

DP amplification is a technique to enhance DP protection based on the intuition that for a sequence of DP mechanisms, if each DP mechanism only acts on part of the data, then for the whole dataset, the corresponding global privacy protection level of the mechanism is stronger than that described in part of the data. To implement DP amplification, sampling is a widely used algorithmic technique, which first randomly samples the data, and then applies a DP mechanism on the randomly selected subset. Intuitively, privacy amplification by sampling is caused by the fact that an individual record has complete privacy if it is not included in the sampled data. For example, if a token never occurs in the training procedure, the attacker will not able to recover it from the encoded information. Therefore, one can construct the relation between sampling probability and privacy cost by exploring the impact on global data protection when applying the DP mechanism on part of the data. This results in a more precise characterization of the privacy protection level while substantially reducing the required noise amount. The most common sampling methods for DP amplification are Reshuffle sampling and Poisson sampling.

Reshuffle Sampling. After shuffling the whole dataset, the quantity of data for each mini-batch is selected from the dataset for calculation, and then the DP mechanism is applied to the computation results. This sampling method is a common practice in neural network training. For each training epoch, the dataset is reshuffled, and an iterator is constructed with a certain batch size. Then each time, a number of samples are taken out in sequence from the shuffled data, and the traversal of the dataset is completed through iteration. To achieve DP amplification, the reshuffle method provides randomization for constructing mini-batch data and inherits the current neural model training process to the greatest extent.
Poisson Sampling. Unlike reshuffle sampling, each sampling process of Poisson sampling is performed independently on the entire dataset. Specifically, if the sampling probability of Poisson sampling is p, for each sampling process, each sample is independently selected from the overall dataset with probability p. Because of the randomness of sampling, compared with reshuffle sampling, the number of samples obtained by Poisson sampling is not uniform each time. Thus, as each sample selected is an independent Bernoulli experiment, the traversal of each data in the whole dataset cannot be guaranteed. However, in general, the effect of DP amplification obtained by Poisson sampling is slightly better than that of Reshuffle sampling, so we adopt Poisson sampling in our algorithms.

Specifically, since an NLP training stage requires even more than thousands of step updates which generate larger required noise, employing different random operations such as sub/up-sampling [**73, 20**] and dropout [**44**] can reduce privacy cost for the same noise power. The detailed analysis and corresponding noise calibration algorithms are provided in Sec. 4.3.3 and Sec. 4.3.4.

2.3.4. DP Post-Processing

The post-processing property is an excellent property of DP proposed by Dwork et al. [**23**], which aims for ensuring the DP guarantee after performing the specific function on the protected object. Intuitively, if we publish a statistic with a certain level of privacy protection, then if this statistic is processed by a function (which can be random), the new degree of privacy protection of statistics should not be lower than the original statistics. The mathematical definition of post-processing property is as follows:
Definition 7. *(DP Post-Processing) Given a randomized mapping function $\mathcal{F} : \mathbb{R} \to \mathbb{R}'$ and a randomized function $\mathcal{M} : \mathcal{X} \to \mathbb{R}$, if $\mathcal{M}$ satisfies ϵ-DP, then $\mathcal{F} \circ \mathcal{M}$ is also ϵ-DP.*

This desired property ensures the algorithm with the DP mechanism still provides DP protection. For example, if we apply a DP mechanism on a token representation, then the token encoder is also satisfied the DP guarantee.

2.3.5. Rényi Differential Privacy Accounting Method

Aforementioned in Sec. 2.3.2, Rényi differential privacy (RDP) [**53**] is a superior DP accounting method. It is a generalization of (ϵ,δ)-DP that uses Rényi-divergence as a distance metric which is also a relaxation of moment accountant [**5**]. It is widely used for private neural model training and a comparison method for us in experimental sections. In this section, we introduce the necessary knowledge of RDP such as its properties for protection level calibration.

2.3.5.1. Theory Introduction. The definition of RDP is formulated as follows:

Definition 8. *(Rényi Differential Privacy). For any two data inputs $\mathcal{X}, \mathcal{X}'$, a DP mechanism $\mathcal{M}$ is (α, ϵ)-RDP with order $\alpha \in (1, \infty)$ if it satisfies*

$$D_\alpha \left(\mathcal{M}(\mathcal{X}) \| \mathcal{M}\left(\mathcal{X}'\right)\right) \triangleq \frac{1}{\alpha - 1} \log \mathbb{E}_{\theta \sim \mathcal{M}(\mathcal{X}')} \left[\left(\frac{\mathcal{M}(\mathcal{X})(\theta)}{\mathcal{M}\left(\mathcal{X}'\right)(\theta)}\right)^\alpha\right] \leq \epsilon$$

For $\alpha \to \infty$, the RDP reduces to $(\epsilon, 0)$-DP, so a randomized mechanism $\mathcal{M}$ is $(\epsilon, 0)$-DP if and only if for any two inputs $\mathcal{X}$ and $\mathcal{X}'$ it satisfies $D_\infty \left(\mathcal{M}(\mathcal{X}) \| \mathcal{M}\left(\mathcal{X}'\right)\right) \leq \epsilon$.

For $\alpha \to 1$, the RDP notion reduces to a Kullback-Leibler-based privacy notion, which is equivalent to a bound on the expectation of the privacy loss random variable.

Generally, the duality between RDP and standard (ϵ, δ)-DP for any $\delta \geq 0$ is expressed in Lemma 1 as we can use for conversion.

Lemma 1. *(RDP to (ϵ,δ)-DP conversion [**53**]). If a DP mechanism $\mathcal{M}$ obeys (α, ϵ)-RDP, then $\mathcal{M}$ obeys $(\epsilon + \frac{\ln(1/\delta)}{\alpha - 1}, \delta)$-DP for all $0 < \delta < 1$.*

To produce the differentially private outputs, the Gaussian mechanism is an example to provide the privacy guarantee associated with the randomized function $\mathcal{M}$ as Lemma 2.

Lemma 2. *(Gaussian Mechanism with RDP) For $f : \mathcal{X} \to \mathbb{R}$ with sensitivity Δ, the Gaussian mechanism $\mathcal{M}$ by adding Gaussian noise with mean 0 and variance σ^2 satisfies $(\alpha, \alpha\Delta^2/(2\sigma^2))$-RDP.*

As the core parts, the DP composition within the RDP accounting framework can compose naturally by Lemma 3 and the sub-sampling amplification is formulated by Lemma 4.

Lemma 3. *(RDP Adaptive Composition [**53**]). If $\mathcal{M}_1$ that takes dataset as input obeys (α, ϵ_1)-RDP, and $\mathcal{M}_2$ that takes the dataset and the output of $\mathcal{M}_1$ as input obeys (α, ϵ_2)-RDP, then their composition obeys $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.*

Lemma 4. *(RDP Sub-Sampling Amplification[3] [**73**]) Given a dataset of n samples drawn from $\mathcal{X}$ and a DP mechanism $\mathcal{M}$ that takes an input from $\mathcal{X}_m$ for $m \leq n$, let the DP mechanism $\mathcal{M} \circ$ subsample be defined as: (1) sub-sample: sub-sample without replacement m data samples of the dataset (sampling parameter $p = m/n$), and (2) apply $\mathcal{M}$: a randomized algorithm taking the sub-sampled dataset as the input. For all integers $\alpha \geq 2$, if $\mathcal{M}$ obeys*

[3]Please refer to *autodp* library for the computation. `https://github.com/yuxiangw/autodp`

$(\alpha, \epsilon(\alpha))$-RDP, then this new randomized algorithm $\mathcal{M}$ o subsample obeys $(\alpha, \epsilon'(\alpha))$-RDP where

$$\begin{aligned}\epsilon'(\alpha) \leq & \frac{1}{\alpha-1}\log\left(1+p^2\binom{\alpha}{2}\min\left\{4\left(e^{\epsilon(2)}-1\right), e^{\epsilon(2)}\min\left\{2,\left(e^{\epsilon(\infty)}-1\right)^2\right\}\right\}\right. \\ & \left.+\sum_{j=3}^{\alpha}p^j\binom{\alpha}{j}e^{(j-1)\epsilon(j)}\min\left\{2,\left(e^{\epsilon(\infty)}-1\right)^j\right\}\right)\end{aligned} \tag{2.3.4}$$

2.3.5.2. Practical Implementation. In practice, to implement privacy accounting under the RDP accounting framework, the accountant procedure is

(1) Given (ϵ, δ)-DP privacy budget, sensitivity Δ and DP mechanism with corresponding noise power σ^2. Compute the optimal α for (α, ϵ_R)-RDP via Lemma 2.
(2) Compose (α,ϵ_{R}/N)-RDP based on RDP Adaptive Composition Lemma 3.
(3) Achieve DP amplification based on RDP Sub-Sampling Amplification Lemma 4.
(4) Convert (α, ϵ_R)-RDP to (ϵ,δ)-DP base on Conversion Lemma 1.

The above steps show how to calibrate the DP protection level via RDP and we also follow them to conduct the comparison.

2.3.6. μ-Gaussian Differential Privacy Accounting Method

In addition to standard (ϵ, δ)-DP and RDP, the μ-Gaussian Differential Privacy [**20**] (μ-GDP) is proposed as another advanced privacy accounting method with better DP composition and DP amplification effectiveness. Since our models are based on this framework, we will give some necessary background knowledge first on how to achieve μ-GDP as well as its privacy calibration method. More details can be found in the paper [**20**].

2.3.6.1. Theory Introduction.

Hypothesis Testing. The definition of DP points out that we can define the level of privacy protection by measuring the distributional difficulty of distinguishing two data inputs $(\mathcal{X}, \mathcal{X}')$ that differ in one data item after being protected by a random function $\mathcal{M}$. By applying the random function $\mathcal{M}$, we can think that the results generated by two inputs $(\mathcal{M}(\mathcal{X}), \mathcal{M}(\mathcal{X}'))$ obey two probability distributions (D_1, D_2). Dong et al. [**20**] re-examine DP from the perspective of hypothesis testing and consider a hypothesis testing problem as

$$H_0 : \text{the computation results are from } \mathcal{X} \qquad H_1 : \text{the computation results are from } \mathcal{X}'$$

as well as a rejection rule $0 \leq \phi \leq 1$, with type I and type II error rates defined as

$$\alpha_\phi = \mathbb{E}_{D_1}(\phi) \qquad \beta_\phi = 1 - \mathbb{E}_{D_2}(\phi)$$

where $\mathbb{E}_D(.)$ represents the cumulative distribution function of the probability distribution D. The type I error is α error, which refers to the situation where the null hypothesis is actually true, but it is rejected. So it is also called the *false positive*. The type II error is

the β error, which refers to the situation where the null hypothesis is actually false, but it is accepted. So it is also called *false negative*.

f-DP. Based on hypothesis testing, Dong et al. [**20**] introduces f-DP, which is the original form of μ-GDP. A trade-off function is defined based on hypothesis testing error as follows:

Definition 9. *(Trade-off function) For any two probability distributions D_1 and D_2 acting on the same space, their trade-off function $f = T(D_1,D_2) : [0,1] \rightarrow [0,1]$ can be defined as*

$$f = T(D_1,D_2)(\alpha) = \inf_{0 \leq \phi \leq 1} \{\beta_\phi : \alpha_\phi \leq \alpha\} \tag{2.3.5}$$

where β_ϕ and α_ϕ are two type error rates defined before and the α is a statistic to be tested range in [0, 1].

The trade-off function describes the relation between the two error types in hypothesis testing. With the trade-off function defined, the relation between DP and hypothesis testing can be established. Then the f-DP is defined as

Definition 10. *(f-DP) For a trade-off function f and two arbitrary data input $\mathcal{X}$ and $\mathcal{X}'$ that differ in one data item, a random mechanism M satisfies f-DP if it satisfies the following condition*

$$T(M(\mathcal{X}),M(\mathcal{X}')) \geq f \tag{2.3.6}$$

Meanwhile, Dong et al. [**20**] point out that f-DP is a generalized version of (ϵ,δ)-DP and the conversion relation between f-DP and (ϵ,δ)-DP can be established by

$$f_{\epsilon,\delta}(\alpha) = max(0,1 - \delta - e^{\epsilon}\alpha,e^{-\epsilon}(1 - \delta - \alpha)) \tag{2.3.7}$$

The f-DP provides a new perspective of DP. Through the trade-off function, we can understand DP more comprehensively, and the DP composition and DP amplification properties can be transformed into the trade-off function. The f-DP is a general analysis framework, which can analyze any DP mechanism. In the following, we will introduce μ-GDP, an instance mechanism with Gaussian based on f-DP.

μ-GDP. μ-GDP is a dual representation of (ϵ, δ)-DP with the Gaussian mechanism. Borrowing the idea of hypothesis testing in f-DP, a trade-off function is introduced to describe the privacy protection level. When replacing the trade-off function f in Definition 10 with a trade-off function for hypothesis testing of two Gaussian distributions, the privacy protection level can be transformed into the difficulty of distinguishing two Gaussian probability distributions. Thus, Dong et al. [**20**] defines the function G_μ to represent the trade-off function for distinguishing Gaussian distributions $G_1 = \mathcal{N}(0,1)$ and $G_2 = \mathcal{N}(\mu,1)$ corresponding to hypothesis testing as

$$G_\mu := T(G_1 = \mathcal{N}(0,1), G_2 = \mathcal{N}(\mu,1)) \tag{2.3.8}$$

where the $\mu \geq 0$ is a duality parameter.

The closed-form solution of the above trade-off function G_μ can be derived according to the known density function and cumulative distribution function of the Gaussian probability distribution as

$$G_\mu(\alpha) = \Phi(\Phi^{-1}(1-\alpha) - \mu) \tag{2.3.9}$$

where Φ represents the cumulative distribution function of the Gaussian distribution and μ can also be understood as the privacy parameter under this framework. Similar to ϵ, μ is negatively correlated with the privacy protection level. When $\mu = 0$, the two distributions that need to be distinguished completely overlap, achieving complete privacy protection, but meanwhile losing usability. When μ is too large, it becomes very easy to distinguish two distributions, and the corresponding privacy protection strength is also very weak. Further, μ-GDP can be defined as a Gaussian distribution version f-DP as follows:

Definition 11. *(μ-Gaussian Differential Privacy) A DP mechanism $\mathcal{M}$ is said to satisfy μ-Gaussian Differential Privacy (μ-GDP) if it satisfies the following condition*

$$T\left(\mathcal{X}, \mathcal{X}'\right) \geq G_\mu \tag{2.3.10}$$

The Eq. 2.3.10 instantiates f-DP as μ-GDP with the duality parameter μ and combines DP protection level with Gaussian distribution hypothesis testing. Therefore, μ-GDP can achieve the desired privacy protection level by injecting calibrated Gaussian noise into the object we want to protect. Similar to f-DP, the duality between the original DP privacy cost parameter (ϵ,δ) and the μ-GDP privacy parameter μ should be implemented. Within the Gaussian mechanism, if z is a random variable following a Gaussian distribution which satisfies $z \sim \mathcal{N}\left(0, \sigma^2 I_d\right)$ and $f : \mathcal{X} \to \mathbb{R}^d$ is the function applied on the data, the conversion between μ-GDP and standard (ϵ, δ)-DP corresponding to Gaussian mechanism $\mathcal{M}(\mathcal{X}) = f(\mathcal{X}) + z$ is formulated as

Lemma 5. *(Conversion between μ-GDP and (ϵ, δ)-DP [**73**]) The duality between μ-GDP and (ϵ, δ)-DP shows that, for all $\epsilon \geqslant 0$, μ-GDP implies $(\epsilon, \delta(\epsilon; \mu))$-DP by*

$$\delta(\varepsilon; \mu) = \Phi\left(-\frac{\epsilon}{\mu} + \frac{\mu}{2}\right) - \mathrm{e}^{\epsilon}\Phi\left(-\frac{\epsilon}{\mu} - \frac{\mu}{2}\right) \tag{2.3.11}$$

where

$$\mu = \frac{\Delta_2}{\sigma} \tag{2.3.12}$$

where Δ_2 refers to the ℓ_2-sensitivity of f and $\Phi(\cdot) = (1 + erf(\sqrt{2}))/2$ refers to cumulative distribution function of standard normal distribution. The other variables are hyper-parameters.

2.3.6.2. Practical Implementation. The above results show that in order to obtain an (ϵ, δ)-DP-protected output based on Gaussian mechanism for a function $f : \mathcal{X} \to \mathbb{R}$ with

sensitivity Δ_2, it is enough to find a noise power variance σ^2 satisfying homologous privacy level. To implement μ-GDP, one should first set up a privacy cost parameter (ϵ, δ) referring to the final privacy level. Then, for each training step t, the privacy cost is computed by the duality Eq. 2.3.11. To reduce the privacy cost, DP amplification can be generated under the μ-GDP framework as

$$\mu_t = \sqrt{\ln\left(\frac{\mu^2}{p^2 t} + 1\right)} \tag{2.3.13}$$

where μ_t indicates the new duality parameter with achieving DP amplification and p denotes the sampling rate. Then, the model training will stop at step T when the privacy cost achieves its threshold. Therefore, for $t \in [1, T]$, all μ_t achieve the DP composition. Finally, the DP composition and amplification are connected.

We exploit μ-GDP as the DP accounting method for our proposed privacy-preserving NLP model because its lower bound of privacy cost is a good approximation and easily converts with (ϵ,δ)-DP. In addition, it is easier to implement privacy protection level calibration with DP composition and amplification compared with RDP. More algorithm details of our method will be discussed in Chapter 4.

2.4. Differentially Private NLP

More and more efforts [**44, 52, 35, 81, 41, 62, 30**] aim to preserve the utility of the text data with provable and quantifiable privacy guarantees via DP. When applying DP in deep learning, there are two common usage settings: Centralized DP (CDP) [**5**] and Local DP (LDP) [**21**], whose overview is depicted in Fig. 2.3.

Centralized DP. The CDP setting is suitable when the central server is considered trustworthy and acts as a centralized data aggregator. The general idea of most existing differentially private NLP methods [**41, 22, 79, 7**] under the CDP setting is to protect the training data by introducing additive randomized noise into the clipped gradients computed from a random group of data samples. For text protection under the CDP settings, Mcmahan et al. [**48**] successfully exploit DP to train a small word-level language model and integrated it into a federated learning framework. More recently, Anil et al. [**7**] and Dupuy et al. [**22**] both show how to efficiently train a privacy-preserving NLP model via DP-SGD [**24**]. Li et al. [**41**] and Yu et al. [**80**] focus on differentially private parameters tuning for better model performance and use re-parameterized gradient perturbation method as well as memory-saving technique to improve the training efficiency. However, the CDP setting still cannot address the privacy concern of the data owners who do not trust any third parties including the central servers.

Local DP. To address the concerns of privacy-conscious users, the LDP setting has been proposed which allows data owners to privatize their data locally before releasing them. It

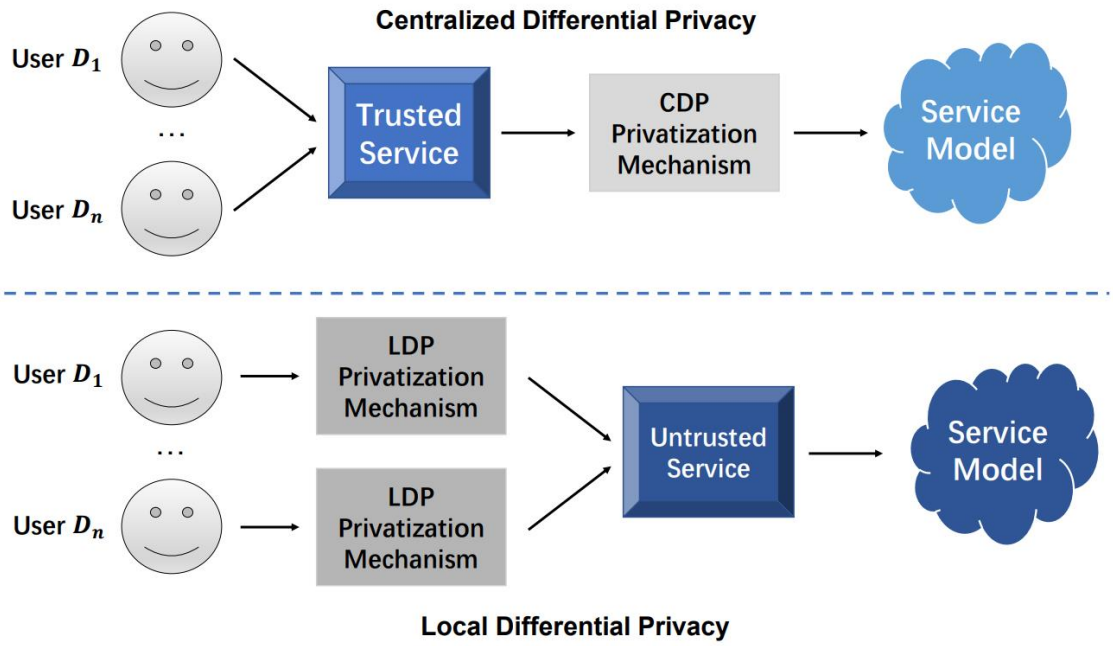


Fig. 2.3. An overview of differences between CDP and LDP.

becomes a pressing problem that remains less explored. For the LDP setting, since privatizing the text data from the root will hurt the semantics and syntax of the sentence, the challenge is how to maintain the utility-privacy trade-off. Some works [**30, 62, 81**] consider a token-to-token privatization and rely on $d_{\mathcal{X}}$-privacy [**17**] definition to preserve the token semantics. Among them, both Feyisetan et al. [**30**] and Qu et al. [**62**] report available model accuracy but with large privacy parameter (e.g., $\epsilon > 50$) or become a random classifier under strong privacy condition (e.g., $\epsilon < 10$), which indicates no practical value. The state-of-the-art SANTEXT method [**81**] ingeniously combines $d_{\mathcal{X}}$-privacy with Exponential mechanism [**49**] to avoid the curse of dimensionality problem. Besides, some LDP works [**44, 38, 35**] privatize text representation before they are collected by untrusted server provider and prevent potential eavesdropped. The crucial procedure is to correctly estimate sensitivity and compose for multiple training steps. However, some open questions still remain. For example, Habernal [**34**] provides detailed proof to argue the sensitivity estimation is wrong in paper [**38**] and the same problem can be found in paper [**44**]. We will explore this problem in our models when producing privatized text representation.

The above CDP and LDP settings define the privacy scenario for corresponding private NLP works. From the utilization point of view, we can further consider how to improve the utility of a specific local text privatization mechanism, which can be connected with customized local DP.

Customized Local DP. The traditional token-to-token LDP mechanisms [**30, 62, 81**] give each data sample the same degree of privacy protection, which jeopardizes the utility due to

the over-protection of data privacy. Since different users have different concerns about their data privacy (e.g. the minority are more concerned about their special information) and the sensitivity of different values is different within the value range (e.g. people are more concerned about writing negative comments online rather than the positive ones), we provide adaptive DP protection for different tokens in text instead of treating them equally. Recently works [**56, 76, 55**] propose new LDP mechanisms that give customized privacy protection for each user's private data and bring a significant boost in the utility. The main idea of them is to treat different sensitive data with unequal protection strength. The sensitive category classification is based on (user) data distribution estimation. Though their methods are applicable for user attribute data but not unstructured text, we can provide adaptive protection for various data by semantic similarity. Inspired by them, our customized text privatization mechanism CusText in Chapter 3 provides a new way of designing customized LDP in the text domain.

2.5. Threat Model

The *Threat Model* is an academic term in the privacy field that describes what information is available for the attacker. Following the previous works [**62, 30, 81**], we consider a *semi-honest threat model* [**28**] under the LDP setting where data owners only submit privatized texts or privatized representations to service providers. Malicious service providers may try to learn sensitive information from their received information. We assume adversaries only have access to the privatized information, and all algorithms/mechanisms are publicly known. Besides, we assume the adversaries have unlimited computation resources. Our privacy-preserving methods and privacy-utility experiments are conducted under the above conditions.

In this chapter, we described the main concepts and techniques for defending privacy risks in NLP, especially the core foundation of our method: differential privacy. Among them, the DP mechanisms are used to establish privacy protection. The advanced DP accountant framework with composition and amplification is used for calibrated protection level and to achieve better privacy-utility trade-off. This preliminary background knowledge helps us to design two text privatization methods CusText and GauDP in the following chapters, which are more sophisticated compared with existing approaches.

Chapter 3

CusText: A Customized Text Privatization Mechanism with Differential Privacy

3.1. Introduction

To protect the privacy of the data owners, a typical way is to privatize their text data locally before releasing them to the NLP service provider for further applications. In existing text privatization mechanisms [**30, 62, 81, 29, 75**], the privacy of the input text is usually guaranteed by Differential Privacy (DP) [**24**] or its variants (e.g. d_χ-privacy), which ensures data privacy with calibrated perturbation. Among them, SANTEXT [**81**] has demonstrated to be the state-of-the-art approach and greatly improved the efficiency of the text privatization process compared with other mechanisms [**30, 29, 75, 62**].

The basic idea of SANTEXT is to generate privatized texts by replacing the original tokens in the text sequentially with new tokens, which are generated by DP guarantee. Despite its effectiveness, SANTEXT faces two inherent limitations. First, satisfying d_χ-privacy limits the applicability of SANTEXT for some similarity metrics such as cosine similarity [**54**] and TF-IDF [**66**]. Second, SANTEXT cannot achieve a good privacy-utility trade-off, i.e., either a large privacy cost with insufficient protection or a small privacy cost with unsatisfiable model accuracy. The first limitation is caused by the definition of d_χ-privacy since it tries to give adaptive privacy protection based on the distance between the tokens, which is currently only applicable to Euclidean distance metrics. The second limitation arises as SANTEXT treats each token in the text equally by assigning each input token with the same output set, which is excessively large (e.g., the output set size could be over 80000). The input token set and output token set denote the two $\mathcal{X}$ and $\mathcal{Y}$ of the score function in Exponential mechanism as described in Definition 6. Such a large output set will lead to the over-protection of input text and hurts the model's utility. Intuitively, to resolve the first limitation, we can convert the proposed privatization mechanism to satisfy ϵ-DP rather than d_χ-privacy. Because under the ϵ-DP notion, one can use any similarity metric

measuring two input tokens. But the difficult point is how to keep the adaptive privacy protection as proposed in d_χ-privacy i.e. still keep the semantic information between tokens. To tackle the second limitation, we can assign each input token a customized output set of a smaller size. What we should deal with is how to customize the output set for each input token that achieves a good privacy-utility trade-off needs to be explored. Thus, resolving these limitations is non-trivial. It is a challenging task to design an effective mechanism that retains the advantages of d_χ-privacy and satisfies ϵ-DP notion for wider adaptability at the same time. Our goal is to privatize raw text locally via customized DP protection that meets ϵ-DP based on the Exponential mechanism.

3.2. Overview

Here, we aim to design Customized Text privatization mechanism named *CusText* that provides more advanced adaptive privacy protection at the token-level. The CusText is applicable to all similarity metrics by turning the mechanism from satisfying d_χ-privacy to satisfying ϵ-DP as we give the proof in Appendix A.1. Meanwhile, CusText inherits the merits of d_χ-privacy by designing a proper *score function* to overcome the shortcoming of the ϵ-DP notion that it cannot provide adaptive privacy based on the semantic similarity between the tokens. The score function is also available for the Exponential mechanism to achieve DP protection. Furthermore, we assign each input token a customized output set of a relatively small size to achieve better adaptive token-level privacy protection. Its relatively small size enables CusText to sample output tokens that are more semantically related to their corresponding input token, thus alleviating the over-protection problem and retaining relatively better performance on downstream tasks.

Conceptually, the sampling process is performed based on a given mapping. Three types of mappings ranging from aggressive to conservative are provided to assign the customized output set for each input token to satisfy different privacy protection needs, i.e., the trade-off between utility and privacy. The utility-privacy trade-off in CusText can be further adjusted by a customization parameter K, which determines the size of the output set for each input token. In addition to the existing token-level text privatization strategy, we propose two more privatization strategies with a larger granularity, i.e., at the record-level or text-level, to reduce the entropy of the sampling distribution, which further improves the utility of the privatized text. Furthermore, since not all tokens contain sensitive information, the above three text privatization strategies which replace all tokens might be over-protective. Therefore, we can retain some original tokens that have low privacy risk (e.g., stopwords) to improve the utility of the privatized text. Skipping some tokens in the raw text can improve the efficiency of the text privatization process as well.

Problem Formulation. We formulate our text privatization task as follows: suppose each document $D = \langle R_i \rangle_{i=1}^{m}$ contains multiple records[1] R and each record $R = \langle t_j \rangle_{j=1}^{n}$ contains multiple tokens t. Given an input text D that contains sensitive information, a global input set $\mathcal{X}$ and a global output set $\mathcal{Y}$ contains all potential input and output tokens, and a text privatization mechanism $\mathcal{M}$, we consider a *token-to-token* case where each token $t_j \in D$ is privatized with $\mathcal{M}$ to get its corresponding privatized token t'_j sampled from $\mathcal{Y}$ if $t_j \in \mathcal{X}$. Then the privatized tokens forms the privatized text $D' = \langle R'_i \rangle_{i=1}^{m}$.

Method Overview. A high-level overview of our customized text privatization mechanism CusText is presented in Fig. 3.1. In general, Custext aims to replace the original text with a new text to achieve the privacy goal. It mainly consists of three components: (1) a mapping function $f_{\mathsf{map}} : \mathcal{X} \rightarrow \{\mathcal{Y}' \subseteq \mathcal{Y}\}$ which determines the output set $\mathcal{Y}'_j$ for each input token $x_j \in \mathcal{X}$ based on a semantic metric; (2) a sampling function[2] $f_{\mathsf{sample}} : \mathcal{X}' \rightarrow \mathcal{Y}'$ based on the Exponential mechanism, to sample a new token from an output set to privatize the input token; (3) a text privatization strategy $\mathcal{S}$, to give instructions when privatizing the text i.e. make the repeated token be mapped to the same token. Specifically, under a text privatization strategy $\mathcal{S}$, for each $t_j \in D$, CusText first gets the output set $\mathcal{Y}'_j$ corresponding to t_j by f_{map}, i.e., $\mathcal{Y}'_j = f_{\mathsf{map}}(t_j)$, then f_{sample} samples an output token t'_j from $\mathcal{Y}'_j$ as the privatized token of t_j, i.e., $t'_j = f_{\mathsf{sample}}(t_j)$, $t'_j \in \mathcal{Y}'_j$. Finally, after applying CusText on each input token t_j in D according to the text privatization strategy $\mathcal{S}$, the final D' is formed by all output privatized tokens. An example that does not replace stopwords privatized by CusText is provided below:

Original Text: when did **spielberg** and **irving marry**?

Privatized Text: when did **scenario** and **treasure mademoiselle**?

3.3. Methodology

3.3.1. Mapping Function

Within text privatization mechanisms, the mapping function $f_{\mathsf{map}} : \mathcal{X} \rightarrow \{\mathcal{Y}' \subseteq \mathcal{Y}\}$ produces the output set for each input token. In SANTEXT [**81**], different input tokens share the same output set while we improve it in CusText by making different input tokens may have different output sets. Then, if a bunch of input tokens in global $\mathcal{X}$ are mapped to the same output set $\mathcal{Y}'$, they belong to the same input set $\mathcal{X}'$ ($\mathcal{X}' \subseteq \mathcal{X}$). The comparison of the mapping function between CusText and SANTEXT is shown in Fig. 3.2.

[1]Could be a sequence sample in NLP dataset.
[2]Given $\mathcal{Y}' \subseteq \mathcal{Y}$, $\mathcal{X}' = \{x | x \in \mathcal{X}, f_{\mathsf{map}}(x) = \mathcal{Y}'\}$.

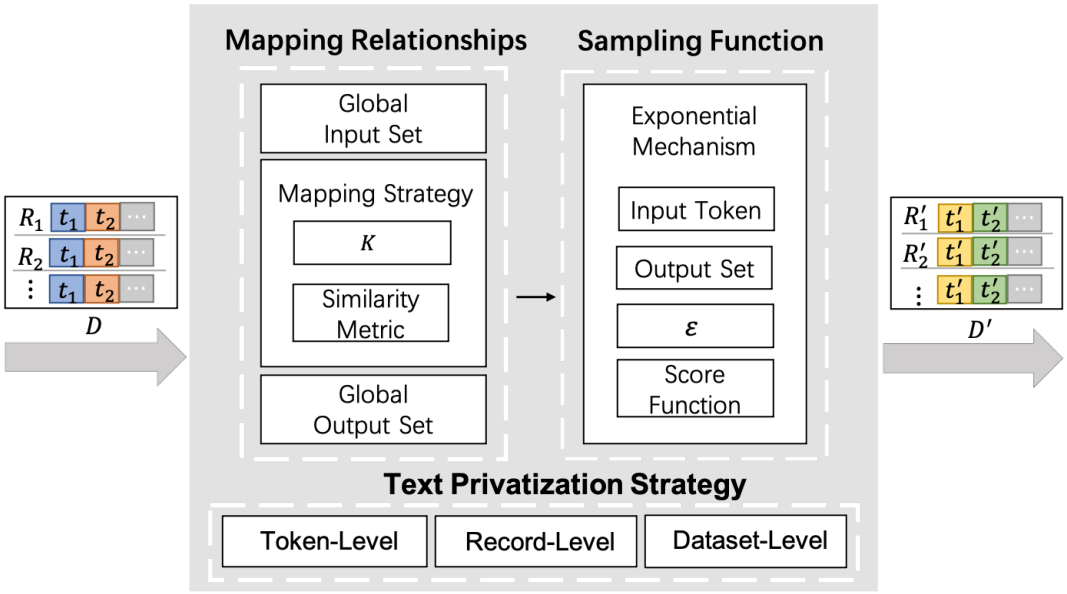


Fig. 3.1. The Overview of CusText.

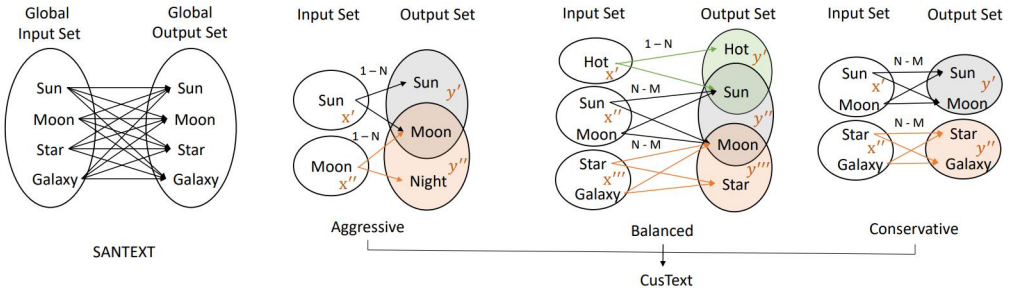


Fig. 3.2. The comparison of mapping function between SANTEXT and our CusText. The figure only contains some core examples within three mapping strategies ($K = 2$), but not the complete mapping relations. Each circle indicates a token set.

When using CusText, the mapping function f_{map} needs to be pre-determined. According to the mapping relation $f_{\mathsf{map}} : \mathcal{X} \to \{\mathcal{Y}' \subseteq \mathcal{Y}\}$, for each input token $x \in \mathcal{X}$, we can find which output set $\mathcal{Y}' = f_{\mathsf{map}}(x)$ it is mapped to and which input set $\mathcal{X}' = \{x | x \in \mathcal{X}, f_{\mathsf{map}}(x) = \mathcal{Y}'\}$ it belongs to. Based on the size of $\mathcal{X}'$ and the size of $\mathcal{Y}'$, we categorize the input token into four types: 1 - 1, N - 1, 1 - N and N - M (1, N, M denote the size of the input/output set and N, M $>$ 1). In Fig. 3.2 we illustrate some general cases about the input tokens belonging to the type N - M and 1 - N with specific mapping functions introduced later. As type N - 1 and type 1 - 1 are not commonly used in existing differentially private text privatization mechanisms, we do not consider them in our methods. Technically, if we want CusText to provide ϵ-DP protection to all input tokens, it requires all input tokens in the global input set $\mathcal{X}$ to belong to type N - M or type N - 1, which ensures that every input token has adjacent tokens. As shown in Fig. 3.2, an input set circle containing more than one token means this bunch of tokens belongs to N - M mapping. Adjacent tokens mean that two data

inputs differ in one data item as in DP Definition 1. This is because the original intention of ϵ-DP is to make the adjacent tokens indistinguishable, so as to protect the input token from being inferred. Our mechanism CusText would contain both type N - M and type 1 - N.

The generation of $f_{\mathsf{map}} : \mathcal{X} \rightarrow \{\mathcal{Y}' \subseteq \mathcal{Y}\}$ is established by assigning the output set for each input token under a mapping strategy, which takes the semantic closeness under consideration. We provide three mapping strategies to generate f_{map} in CusText for different scenarios with the examples shown in Fig. 3.2. From the privacy perspective, we hope to contain as many token mappings belonging to type N - M as possible, because the type 1 - N without adjacent tokens cannot guarantee DP protection. These three mapping strategies approach this goal to vary degrees. For example, the conservative mapping contains the most type N - M, while the aggressive contains the most type 1 - N. However, the number of type 1 - N token mappings would affect the downstream task utility. For simplicity, we unify the size of each input token's corresponding output set $\mathcal{Y}'$ to K and define K as the customization parameter. The details for generating f_{map} are presented in Algorithm 1 with the balanced mapping as the default mapping strategy. For every token x that needs to be privatized, we first calculate the semantic distance between each candidate token y in the output set and the input token x and select the K closest ones. Then we generate the mapping function according to the mapping strategy.

Assuming we are processing the token *marry* within the given original text as below and the top-3 semantically closest tokens of it are {*married, engage, divorce*}. Jointly with the description in Fig. 3.2, we give some intuitive examples for different mapping results.

Original Text: when did spielberg and irving **engage** and **marry**?

For aggressive mapping, we directly assign the candidate set as the output set for x and add x to the temporary input set. Thus, the output set could be the top-k semantically closest tokens to the *marry* in the whole vocabulary, that are {*married, engage, divorce*}. And the *marry* will be added to the temporary input set. So far, the temporary input set contains all tokens in the Original Text sentence. For balance mapping, we assign the tokens that are in the candidate set but not in the temporary input set with the candidate set as the output set. As the result, the top-3 semantically closest tokens to *engage* (e.g. *appointment*, *date* and *marry*) which also do not occur in the temporary input set will be added to the output set. The closest token *marry* will be eliminated because it is in the temporary input. And as we only keep the top-3 (assume *appointment* and *divorce* are the 4^{th} and the 5^{th} token close to *marry*), the final output set of balance mapping could be {*married*, *engage*, *date*}. For conservative mapping, we eliminate the tokens in the candidate set from the output set base on the balanced mapping. Thus, the token *engage* will be eliminated as it has occurred in the previous output set and the final output set for conservative mapping could

Algorithm 1 Generating Mapping Function

Input: Customization parameter K, distance function f, mapping mode M, input set $\mathcal{X}$, output set $\mathcal{Y}$

Output: Mapping function f_{map}

for $token \in \mathcal{X}$ **do**
  Initial candidate set $L_{token} = \varnothing$, distance set $d = \varnothing$, temporary input set $T = \varnothing$
  for $token' \in \mathcal{Y}$ **do**
    Calculate semantic distance $d_{token'} = f(token, token')$
    Add $d_{token'}$ to d
  end for
  Add K smallest $d_{token'} \in d$ of its responding $token'$ to L_{token}
  if M = Aggressive **then**
    Assign $L_{token} \rightarrow f_{\mathsf{map}}[token]$
    Add $token$ to T
  else
    for $token' \in L_{token}$ **do**
      if $token' \notin T$ **then**
        Assign $L_{token} \rightarrow f_{\mathsf{map}}[token']$
      end if
    end for
    if M = Conservative **then**
      Assign $(\mathcal{Y} - L_{token}) \rightarrow \mathcal{Y}$
    end if
  end if
end for
return f_{map}

be {*married*,*date*}. The mathematical form for these three mapping functions is described below.

- **Aggressive Mapping.** For each input token $x \in \mathcal{X}$, it leverages a certain similarity metric to select K tokens $y \in \mathcal{Y}$ which are semantically closest to x as its customized output set. Such mapping strategy makes most of the tokens in $\mathcal{X}$ belong to type 1 - N, which means few of them will have adjacent tokens.
- **Balanced Mapping.** Based on the aggressive mapping, the balanced mapping tries to make more input tokens belong to type N - M by mapping more than one input token to the same output set, i.e., for most of the input tokens $x \in \mathcal{X}$, $\exists\ x' \in \mathcal{X}$ and $x' \neq x$ s.t. $f_{\mathsf{map}}(x') = f_{\mathsf{map}}(x)$. However, under the balanced mapping, for most input tokens x,$x' \in \mathcal{X}$, if $f_{\mathsf{map}}(x') \neq f_{\mathsf{map}}(x)$, we have $f_{\mathsf{map}}(x) \cap f_{\mathsf{map}}(x') \neq \emptyset$.
- **Conservative Mapping.** Based on the balanced mapping, the conservative mapping makes all input tokens in $\mathcal{X}$ belong to type N - M by making different output sets $\mathcal{Y}' \subseteq \mathcal{Y}$ to have no intersections, i.e., $\forall\ x$,$x' \in \mathcal{X}$, if $f_{\mathsf{map}}(x') \neq f_{\mathsf{map}}(x)$, we have $f_{\mathsf{map}}(x) \cap f_{\mathsf{map}}(x') = \emptyset$.

3.3.2. Sampling Function

Based on the mapping function $f_{\mathsf{map}} : \mathcal{X} \rightarrow \{\mathcal{Y}' \subseteq \mathcal{Y}\}$, the sampling function $f_{\mathsf{sample}} : \mathcal{X}' \rightarrow \mathcal{Y}'$ can obtain the output set $\mathcal{Y}'$ corresponding to each input token in $\mathcal{X}'$ for implementation. In CusText, we adopt the Exponential mechanism as our sampling function. However, we need to design a suitable score function for the Exponential mechanism to provide adaptive privacy protection at token-level under the ϵ-DP notion.

Two rules should be observed when designing the score function $u(\cdot,\cdot) : \mathcal{X}' \times \mathcal{Y}' \rightarrow \mathbb{R}$:

(1) The score for each input and output token pair is bounded, i.e., $\forall\ x \in \mathcal{X}'$, $\forall\ y \in \mathcal{Y}'$, $\exists\ M \in \mathbb{R}$ s.t., $u(x,y) < M$.

(2) The higher the semantic similarity between the input token and the output token, the higher the score of $u(x,y)$, i.e., $\forall\ x \in \mathcal{X}'$, $\forall\ y,\ y' \in \mathcal{Y}'$, if $u(x,y) > u(x,y')$, y is semantically closer to x than y'.

Since the ϵ-DP requires the *sensitivity* Δu to be bounded, the first rule helps the Exponential mechanism to satisfy ϵ-DP as we will illustrate in detail with the sampling procedure later. The second rule ensures the candidate tokens that have closer semantics to the input token have higher probabilities to map, so as to retain the advantage of d_χ-privacy.

When designing the score function, we use pre-trained embeddings as the token representations, such as Word2Vec [**51**], GloVe [**59**] and Counter-fitting [**54**]. The similarity metric used in our mechanism should be determined by the embedding type. For instance, we can use Euclidean distance and cosine similarity as similarity metrics for GloVe and Counter-fitting, respectively. Based on the correlation between the similarity score and the semantic similarity, all text similarity metrics could be categorized into two types: *negative correlation* (e.g. the lower the value, the stronger the correlation) and *positive correlation* (e.g. the higher the value, the stronger the correlation). For example, the Euclidean distance belongs to the type negative correlation while the cosine similarity belongs to the type positive correlation. To verify our mechanism can suit different similarity metrics, we provide a possible score function for both similarity metric types.

Negative Correlation. We take Euclidean distance for example to design the score function $u(\cdot,\cdot) : \mathcal{X}' \times \mathcal{Y}' \rightarrow \mathbb{R}$. For any input set $\mathcal{X}'$ and its corresponding output set $\mathcal{Y}'$, we first calculate the Euclidean distance between one input token $x \in \mathcal{X}'$ and each output token y in $\mathcal{Y}'$ to get the distance list $K_d \in \mathbb{R}^{|\mathcal{Y}'|}$. The distance between the input and output token pair is $d(x,y) = \|\Phi(x) - \Phi(y)\|_2$. The $\Phi(x)$ and $\Phi(y)$ denote the embeddings of x and y. Then, we normalize the value of distance list K_d to be ranged in [0,1] by Eq. 3.3.1 and transform the distance list K_d into the score function for the corresponding input token x by $u(x,\mathcal{Y}') = 1 - K_d$. This transformation enables the input and output token pair (x,y) with higher semantic similarity to having a higher score.

$$K_d = \frac{K_d - \min(K_d)}{\max(K_d) - \min(K_d)} \tag{3.3.1}$$

Finally, for each input token in the input set $\mathcal{X}'$, we repeat the above steps to get the complete utility score function $u(\cdot,\cdot) : \mathcal{X}' \times \mathcal{Y}' \to \mathbb{R}$.

Positive Correlation. We take cosine similarity for example to design the score function $u(\cdot,\cdot)$. For any input set $\mathcal{X}'$ and its corresponding output set $\mathcal{Y}'$, we first calculate the cosine similarity between a input token x and each output token y in $\mathcal{Y}'$ to get the cosine similarity list $K_c \in \mathbb{R}^{|\mathcal{Y}'|}$. The cosine similarity of the input and output token pair is $cos(x,y) = \frac{\Phi(x)^T \Phi(y)}{\|\Phi(x)\|\|\Phi(y)\|}$. Then, we normalize the cosine similarity list K_c to produce the score function $u(\cdot,\cdot) : \{x\} \times \mathcal{Y}' \to \mathbb{R}$ for input token x by Eq. 3.3.2

$$u(\cdot,\cdot) = \frac{K_c - \min(K_c)}{\max(K_c) - \min(K_c)} \tag{3.3.2}$$

Finally, for each token in the input set $\mathcal{X}'$, we repeat the above steps to get the complete utility score function $u(\cdot,\cdot) : \mathcal{X}' \times \mathcal{Y}' \to \mathbb{R}$.

Sampling Procedure. After obtaining the available score function, the sampling function f_{sample} is competent to generate the privatized token t'_j for the input token t_j by adopting the Exponential mechanism. In this sampling procedure, we make f_{sample} satisfy ϵ-DP. For any input set $\mathcal{X}'$ and its corresponding output set $\mathcal{Y}'$, the sensitivity Δu between any two adjacent input tokens $x, x' \in \mathcal{X}'$ is bound to 1 as Eq. 3.3.3 based on our design and the first rule of the score function.

$$\Delta u = \max_{y \in \mathcal{Y}'} \max_{x,x' \in \mathcal{X}'} \|u(x,y) - u(x',y)\|_1 = 1 \tag{3.3.3}$$

The sampling procedure is the core of CusText. Formally, given a privacy parameter ϵ, for $\forall x \in \mathcal{X}', \forall y \in \mathcal{Y}'$, the sampling function $f_{\mathsf{sample}} : \mathcal{X}' \to \mathcal{Y}'$ is ϵ-DP if it satisfies

$$Pr[f_{\mathsf{sample}}(x) = y] = \frac{e^{\frac{\epsilon u(x,y)}{2\Delta u}}}{\sum_{y' \in \mathcal{Y}'} e^{\frac{\epsilon u(x,y')}{2\Delta u}}} \tag{3.3.4}$$

Our method CusText can provide ϵ-DP with the above sampling function based on the Exponential mechanism. The proof is shown in Appendix A.1. Based on this sampling function, each token in the output set constructed by the mapping function will be assigned a probability for replacement.

3.3.3. Text Privatization Strategies

In addition to the widely used token-level text privatization strategy [**30, 62, 81**], we propose two other record-level and dataset-level strategies for CusText to privatize the input text. The details of three strategies with different granularities are listed below.

- **Token-Level.** The idea of the existing token-level strategy is straightforward. For each token in each record of the dataset ($\forall\, t \in R \in D$), we run the sampling function to sample a new token and replace the original token with it.
- **Record-Level.** Within the token-level strategy, the repeated token in the same record might be mapped to different tokens. To preserve the syntax similarity between the raw text D and the privatized text D', we force the repeated token in the same record to be mapped to the same token. For example, if a token occurs more than one time in a sequence, then all of its occurrences will be substituted as the same replacement.
- **Dataset-Level.** Though the repeated token in the same record will be mapped to the same token in the record-level strategy, the same token in different records is still possible to be mapped to different tokens. Therefore, we make the repeated token in the whole dataset be mapped to the same token.

The goal of different text privatization strategies is to preserve the original syntax feature to various degrees for improving the utility of privatized text. This may raise the concern that attacks would become easier if the repeated tokens are replaced by the same tokens. We will evaluate this concern in privacy experiments at Sec. 3.4.5. The granularity of text privatization Strategy can be selected according to practical needs.

3.4. Experiments

3.4.1. Experimental Setup

We choose two datasets from GLUE benchmark [**72**] with privacy implications to demonstrate the effectiveness of CusText. The dataset information is listed below.

- **SST-2** is a popular sentiment prediction dataset for movie reviews, with 67k training sentences and 872 validation sentences.
- **QNLI** is a popular dataset for a sentence-pair classification task, with 105k training samples and 5.4k validation samples.

In our experiments, we first utilize CusText to generate privatized texts, then use those privatized texts to fine-tune the pre-trained language models (PLMs), specifically Bert[3]. Next, we contrast privatized texts and un-privatized texts to evaluate the model's performance loss. In particular, we use Counter-fitting embeddings as the token representation, which is based on cosine similarity to measure the semantic similarity between tokens. When producing the privatized text, both the global input set $\mathcal{X}$ and the global output set $\mathcal{Y}$ in CusText are equal to the vocabulary of Counter-fitting, and out-of-vocabulary (OOV) tokens except the numbers, will be retained. For each downstream task, we set the maximum

[3]We use *bert-base-uncased* from `https://huggingface.co/bert-base-uncased`

sequence length to 128, training epoch to 1, batch size to 64 and learning rate to 2e-5. Other hyper-parameters setups are kept default as the Transformer library [**74**].

Evaluation Metric. We mainly use accuracy as the evaluation metric on the test set with privatized text and it is calculated by

$$\text{Accuracy} = \frac{|\text{correctly predicted samples}|}{|\text{total samples}|}$$

The same evaluation method is applied in the next Chapter for *GauDP* model.

3.4.2. Comparison of Different Text Privatization Mechanisms

We first compare our CusText under different customization parameters K at privacy parameter $\epsilon = 1$ with other text privatization mechanisms. The customization parameter K is to determine the size of the output set for each input token that needs to be privatized, and $\epsilon = 1$ means strong protection in the context of DP. The protection level ϵ is calibrated by Eq. 3.3.4 with a pre-defined value of ϵ. Its value would affect the sampling probability of each token in the output set. The main comparison methods include the state-of-the-art SANTEXT [**81**] and FBDD [**30**], the implementation of which follows its original setting. Besides, the *Random* method denotes sampling a privatized token randomly and *Original* refers to the non-privacy setting. Because the value of K controls the degree of customization, smaller K indicates better adaptation to the customized purpose. When applying CusText to privatize SST-2 and QNLI, we use balanced mapping and record-level text privatization strategy. Table 3.1 shows the accuracy of the different text privatization mechanisms for two datasets. The comparison results further confirm the effectiveness of our customized mechanism CusText under a strong privacy protection condition ($\epsilon = 1$), while the results of the other mechanisms are similar to random replacement. Besides, we observe that the trained model performs better with a smaller K in CusText, this indicates the importance of preserving token semantics when implementing privacy protection. The above results reflect that shrinking the output set for achieving customized DP can indeed mitigate the over-protective problem among existing methods. By controlling K, we can preserve more original semantics for approaching the non-privatized results and yield higher accuracy than other methods without customization.

Apart from the utility, we also analyze the privacy impact brought by different K, by providing some privatized examples as a case study from QNLI under different $K = [5, 50, 200]$ in Table 3.2. We observe that privatized text will be semantically closer to the raw text with a smaller K. This indicates that smaller K may fail to protect the privacy of the raw text though it has a better utility. The results in Table 3.1 and Table 3.2 suggest that neither

Dataset	Random	SANTEXT [81]	FDBB [30]	Original	CusText				
					$K = 5$	$K = 20$	$K = 50$	$K = 200$	$K = 500$
SST-2	0.4986	0.5101	0.5099	0.9163	0.9117	0.8761	0.8383	0.8119	0.7913
QNLI	0.5152	0.5372	0.5163	0.8947	0.7804	0.5665	0.5441	0.5034	0.5000

Table 3.1. Accuracy of various text privatization mechanisms with privacy parameter $\epsilon = 1$.

Original	when did spielberg and irving marry?
	then in 1984 they renewed their romance, and in november 1985, they mairried, already having had a son, max samuel.
$K = 5$	when did hanks and irving marries?
	then in 2811 they renew their ballad, and in nov 2467, they marries, after having had a son, maximum josiah.
$K = 50$	when did theatrcal and benson hens?
	then in 2708 they refitted their modern, and in marked 2218, they daughter, therefore having had a kiddo, paramount jeremiah.
$K = 200$	when did scenario and treasure mademoiselle?
	then in 2702 thet renewed their sweet, and in hsien 2451, they gender, today having had a school, maximizing abram.

Table 3.2. Qualitative examples from QNLI dataset: Privatized text by CusText under different customization parameter K. The privatization is based on the balanced mapping, record-level text privatization strategy with saving stopwords and privacy parameter $\epsilon = 1$.

too big nor too small K, e.g., $K = 50$, will be a good choice to achieve a good utility-privacy trade-off. In practice, the customization parameter K should be carefully selected for different models and datasets.

3.4.3. Comparison of Mapping Strategies

We next compare three mapping strategies on two aspects: (1) the accuracy of the downstream task; (2) the proportion of input tokens that do not belong to type N - M mapping. As we mentioned previously, most type N - M mapping means better privacy because the 1 - N type contained in our methods cannot provide a DP guarantee. Thus, the accuracy and the proportion of input tokens that do not belong to type N - M mapping form the utility-privacy trade-off for different mapping strategies. We conduct the experiments on SST-2 under three different customization parameters $K = [20,50,100]$ with record-level text privatization strategy and privacy parameter $\epsilon = 1$. From the results in Table 3.3, we find that aggressive mapping can provide the best utility for privatized texts. The good performance of the aggressive mapping might be due to its design principle, which aims to

K	Aggressive	Balanced	Conservative
20	87.96	87.61	83.03
50	84.98	83.83	75.92
100	86.24	84.51	53.56

Table 3.3. Comparison of different K and mapping strategies regarding accuracy on SST-2.

K	Aggressive	Balanced	Conservative
20	96.18 %	0.12%	0.00%
50	99.20 %	0.04%	0.00%
100	99.84 %	0.00%	0.00%

Table 3.4. Comparisons of mapping strategies on the proportion of input tokens **NOT** belong to type N - M mapping on SST-2.

give the "best" output set for its corresponding input token. That is to say, our strategy is designed to make the sampling function sample an output token semantically close to the original one. We also find that aggressive and balance mapping is worse at $K = 50$ than at $K = 100$, which does not reflect the advantage of customization. This might be because the 1 - N types contained in these two mappings make them less sensitive to K than conservative mapping. On the other hand, the results of Table 3.4 show the obvious disadvantage of the aggressive mapping that most input tokens do not belong to type N - M mapping. Besides, although the conservative mapping ensures every token in the global input set $\mathcal{X}$ belongs to type N - M, it has a bad performance in the utility. Overall, we think balanced mapping is the best mapping strategy in our experiments because it not only ensures that most of the input tokens belong to type N - M but also offers good utility, thus offering a good trade-off between them.

3.4.4. Comparison of Different Text Privatization Strategies

We then compare the utility of the text privatization strategies with three different levels: token-level, record-level, and dataset-level. A higher-level strategy tends to make the privatized text more similar to a natural language text. The results are shown in Table 3.5. We can see that the record-level and dataset-level are more applicable than the token-level. This indicates that the high-level privatization alignment is useful for preserving semantics. The accuracy improvement is larger when ϵ increases. It means that as privacy protection becomes weak, the more original semantics can be retained by high-level text privatization strategies. Furthermore, we can find that the improvement brought by record-level and dataset-level strategies on QNLI is more significant than SST-2. This may be because QNLI

$\mathcal{S}$	SST-2			QNLI		
	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$
T	84.63	89.33	90.71	50.83	54.52	64.53
R	86.24	**89.45**	**87.50**	52.18	**66.16**	77.54
D	**86.35**	87.96	88.76	**58.10**	53.30	**77.82**
T^+	89.68	89.91	92.09	65.10	71.57	78.35
R^+	89.22	**90.60**	**92.66**	**79.97**	**81.01**	**89.64**
D^+	**90.25**	90.83	90.37	77.73	79.07	85.11
O	91.63			89.47		

Table 3.5. Comparison of accuracy of different text privatization strategies $\mathcal{S}$ on SST-2 and QNLI. T: Token-Level strategy, R: Record-Level strategy, D: Dataset-Level strategy; $^+$: save stopwords; O: original training dataset.

is applied for a natural language inference task and mapping the repeated tokens in the question-answer pair to the same token is crucial for the model to do the right prediction. In addition, the experiment results also show that saving stopwords could boost the utility under the same privacy guarantee. This may be because saving stopwords helps to preserve the original syntax feature. It is an encouraging finding since it enables us to get a relatively good performance of a downstream task with smaller privacy costs.

3.4.5. Privacy Calibration

To further explore the privacy protection level brought by our text privatization mechanisms, we conduct some supplementary experiments to better empirically calibrate privacy. **Proportion of Original Tokens**. Under the DP mechanism, the proportion of the original tokens in the privatized text does not have a direct connection with differential privacy calibration since the adversaries cannot identify the original tokens. However, when most tokens in the privatized text are original tokens, the privatized text will have high readability as the original text. In such a situation, with the help of human reasoning ability, the original text can be easily inferred and thus fail to achieve privacy protection.

The CusText might overcome this issue because the privatized text produced by CusText is made up of a bunch of independent privatized tokens. The likelihood that the privatized text is still linguistically inferable will be very low. Based on the above discussion, we use the proportion of original tokens as one of the metrics to empirically evaluate the privacy protection provided by text privatization mechanisms. Following the same parameter setting as in Sec. 3.4.4, we evaluate the proportion of original tokens in different situations. Table 3.6 show that the dataset-level privatized strategy can provide better privacy with regard to whether the original tokens occur again in the privatized text than the other two. But

$\mathcal{S}$	SST-2			QNLI		
	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$
T	**2.82%**	9.92%	30.31%	2.82%	9.76%	29.24%
R	2.83%	9.87%	30.29%	2.84%	9.77%	29.26%
D	3.15%	**7.81%**	**30.06%**	**1.36%**	**7.59%**	**16.03%**
T^+	43.52%	47.37%	57.55%	45.85%	49.58%	59.44%
R^+	43.51%	47.33%	57.50%	45.86%	49.58%	59.41%
D^+	43.32%	48.15%	56.75%	45.38%	47.28%	50.79%
O	100.00%			100.00%		

Table 3.6. The proportion of original tokens preserved in the privatized text under customization parameter $K = 50$. A **lower** proportion indicates better privacy protection.

$\mathcal{S}$	SST-2			QNLI		
	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$	$\epsilon = 1$	$\epsilon = 5$	$\epsilon = 10$
T	13.54%	15.33%	24.44%	12.68%	14.40%	23.95%
R	12.26%	16.03%	**24.18%**	**11.56%**	**13.68%**	**23.29%**
D	**11.88%**	**14.88%**	24.66%	12.04%	16.32%	26.13%
T^+	26.57%	31.33%	39.70%	26.64%	30.88%	37.69%
R^+	28.59%	32.32%	31.11%	29.30%	31.67%	38.45%
D^+	27.37%	31.62%	40.18%	27.53%	31.68%	39.06%
O	61.93%			61.93%		

Table 3.7. The percentage of tokens that are successfully inferred by the mask token inference attack. A **lower** percentage indicates better privacy protection.

the larger ϵ and preserve stopwords results in a higher proportion of original tokens which might hurt privacy in this evaluated method. These results confirm that smaller ϵ can bring better privacy protection and help us how to choose ϵ for a suitable proportion of original tokens in the privatized text. Since the proportion of original tokens among the three text privatization strategies is not that significantly different, we think that the proportion of preserved original tokens is not sensitive to the level of privatization strategies, thus we can use a more aggressive strategy to boost the model utility.

Mask Token Inference Attack. We also adopt the same attack experiment as previous work [**81, 62**] to empirically evaluate the privacy protection provided by CusText. In order to recover the original text D from the privatized text D', we assume the adversaries could use the PLMs[4] to infer the original tokens since it is trained via masked language modeling and very popular as an encoder in most NLP downstream tasks. Specifically, the adversaries

[4]We use *bert-base-uncased* for all attack experiments.

could first use the special token [MASK] to replace one token in the privatized text, then input the masked text into the PLM to get the prediction of the [MASK] token and consider the prediction token as the original one. Finally, the whole recovered text is gained by replacing each token in the privatized text sequentially with the above procedure. We follow the same parameter setting in Sec. 3.4.4 to perform the mask token inference attack. The experiment results are shown in Table 3.7, which shows the dataset-level text privatized strategy brings better privacy for SST-2 while the record-level is more applicable for QNLI. Combined with the results in Table 3.6, we observe that although the proportion of original tokens in the privatized text is smaller when ϵ is small ($\epsilon = 1$ or 5), adversaries could recover more than 10% original tokens based on the mask token inference attack, while with $\epsilon = 10$ and retaining stopwords, the quantitative relationship shows the opposite trend. This indicates that the proportion of original tokens and mask token inference attacks provide two empirical perspectives for privacy calibration and they do not seem to have an obvious connection. Our proposed high-level text privatized strategy can achieve higher accuracy and better empirical privacy protection. The retention of stopwords preserves more syntax features for better utility but might also become easier for inference attacks.

Privacy Concern: Can $\mathcal{X} \subseteq \mathcal{Y}$? In both previous works [**30, 62, 81**] and our mechanism CusText, the global input set $\mathcal{X}$ is included in the global output set $\mathcal{Y}$ ($\mathcal{X} \subseteq \mathcal{Y}$). Thus, the privatized text produced by those mechanisms may retain some original tokens as illustrated in Table 3.6. Intuitively, this might lead to privacy leakage because of those unchanged tokens. However, in a practical situation, combining the experiment results of mask token inference attack in Table 3.7 and the proportion of original tokens in Table 3.6, we observe that $\mathcal{X} \subseteq \mathcal{Y}$ has little impact on the privacy protection of the raw text when the proportion of original tokens in the privatized text is low and the protection level ϵ is small.

3.5. Conclusion

In this chapter, we study how to achieve better utility on the privatized text by designing a Customized differentially private Text privatization mechanism (CusText) that provides adaptive privacy protection at the token-level. Specifically, we propose a novel sampling function by designing a suitable score function on top of the Exponential mechanism and providing each input token its own customized output set to boost the utility of privatized text. Meanwhile, it makes the CusText satisfying ϵ-DP notion with broad applicability. Moreover, we provide two new text privatization strategies to improve the utility of privatized text without compromising privacy and evaluate privacy calibration empirically from two views. Extensive experiments show that CusText achieves a better privacy-utility trade-off and has better practical application value.

Chapter 4

GauDP: A Gaussian-based Local Differentially Private NLP Model

4.1. Introduction

The privacy-utility trade-off is the primary issue [**23**] in the context of DP-NLP [**31**]. The reason is that some DP mechanisms such as the Laplace mechanism and Gaussian mechanism invariably significantly reduce the downstream task performance by injecting noise into the text representation for DP protection. These paradigms are different from the Exponential mechanism as we introduce in the last Chapter.

Here, we focus on the practical situation in which the user (data owner) could be an individual or an institution involved in multi-party computation, are concerned about the privacy of their sensitive data and the server provider is untrustworthy. Since the data privacy will still be leaked by model or gradient by specific attacks [**67, 15**], LDP is required to protect the user's input text before sharing their computation results (text representations) with the server provider. As the downstream task is implemented at the server, the user himself cannot perform DP-SGD-based model training to protect the privacy of the training data. Besides, most existing methods [**7, 30, 35, 75, 41**] can only provide protection for the training data but not consider the inference ones. The reason is that they assume the DP guarantee procedure for training and inference are the same, as they did not apply DP composition and amplification. As we introduced in Sec. 2.3, DP composition is necessary for calibrating accurate required noise amount to protect the whole dataset and DP amplification technique (e.g. sub-sampling) can help reduce the amount of required noise for generating better utility through the model's inherent randomness. Without DP composition, the private model might not achieve the protection level as strong as their declaration and less practical value. However, protecting inference data privacy is more difficult because the sub-sampling technique for DP amplification in the training stage cannot be directly applied in the inference stage. For example, the sub-sampling can be implemented directly when

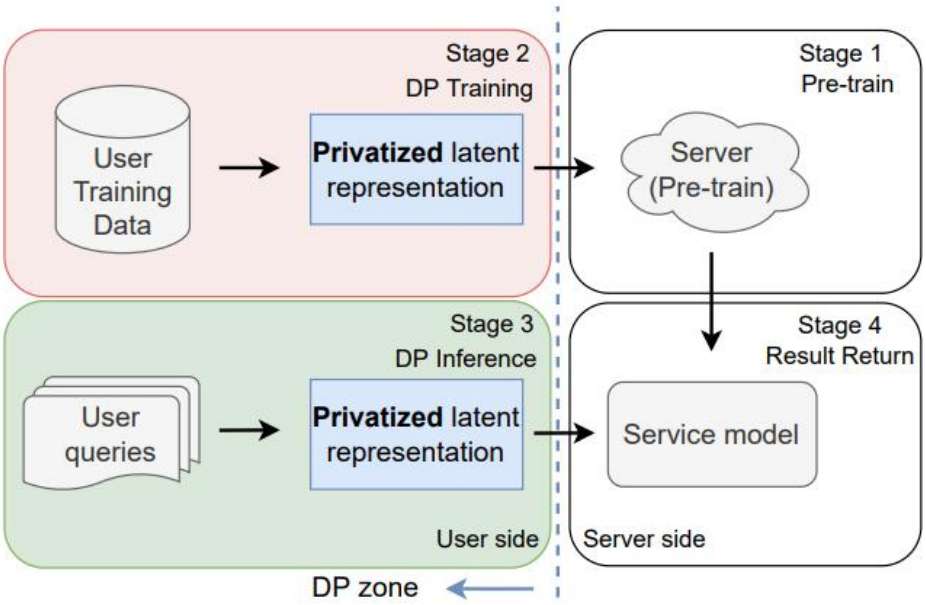
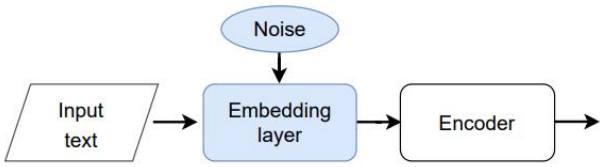


Fig. 4.1. LDP-NLP pipeline for DP training and DP inference in Stages 2 and Stage 3 to protect training and inference.

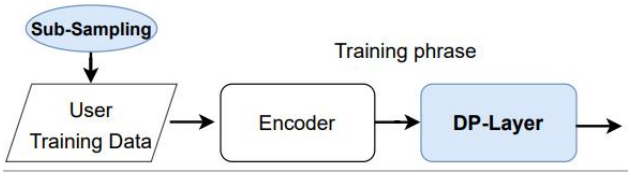
constructing arbitrary training batches, but during inference, all user input needs to be given responses. Thus, solving the aforementioned problems is difficult. To this end, a carefully designed noise calibration algorithm for both training and inference stages with DP composition and DP amplification is necessary to push the privacy-utility trade-off boundary, while there are fewer explorations on it. Our goal is to design a Gaussian mechanism-based model to produce differentially private latent representations locally within the μ-GDP framework for DP accounting under a practical scenario.

4.2. LDP-NLP Task Pipeline

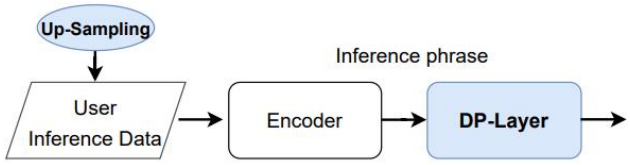
Our goal is to design privacy for NLP tasks from the user side satisfying the LDP setting. As the original user's input contains sensitive information, our primary requirement is that the service provider only accesses LDP-guaranteed representation, which means that all of the information sent out locally by the user in all training and inference steps must satisfy the DP definition, then the adversary would be unable to distinguish the input data. Fig. 4.1 depicts a target scenario with corresponding LDP operations to complete an LDP-NLP task between the user and the server provider, which contains four stages: pre-training, DP training, DP inference, and inference results return. After receiving the DP-guaranteed representations in the training and inference stages, the server computes the back-propagations and forward computations, respectively. Although this study focuses on a single user case, the LDP can help to adapt to a more general setting in which sensitive data is collected independently from multiple users. Fig. 4.2 depicts the differences between existing works on LDP-NLP



(a) Existing work on LDP-NLP [**30, 62, 81**].



(b) Our Sub-Sampling+DP-layer for training. The Stage 2 in Fig. 4.1.



(c) Our Up-Sampling+DP-layer for inference. The Stage 3 in Fig. 4.1.

Fig. 4.2. Different LDP architectures for the privacy-preserving modules (in blue).

and our methods. The benefits of our methods can be viewed from two perspectives. On one hand, the noise for DP protection is injected after the encoder via a DP-Layer which can enable the semantic encoding not to be too much affected by the noise. On the other hand, we leverage the sub/up-sampling technique to implement DP amplification and thus bring a better utility-privacy trade-off by ensuring DP composition compared with existing works. Intuitively, DP amplification by sampling is caused by the fact that an individual sample has complete privacy if it is not included in the sampled set and whether or not this individual sample is included is a secret. Both sampling methods inherited this foundation based on randomness. More details are discussed in the experiments Sec. 4.4. Here, we first give a description of the role of the four stages.

4.2.1. Pre-training (Stage 1)

The concept of word embedding within the neural method is first pioneered by Bengio et al. [**11**] which alleviates the dimensional disaster of the language model. Then the foundation for the subsequent study of word representation learning, such as Word2Vec [**51**], GloVe [**59**] and BERT [**19**] are laid down. Word embedding is a feature learning technique in which each word or phrase from the vocabulary is mapped to a N dimensional vector of real numbers.

Although the real numbers of each dimensional are unreadable to humans, it is machine-perceived and can represent high-level semantics in high-dimensional spaces and often get better performance as input to multi-layer perceptrons (MLP) than the traditional feature extraction methods. For our target scenario, the server performs pre-training on its own such as exploiting large-scale public corpus or domain-specific private data for getting pre-trained word embeddings on stage 1. In our method, we utilize GloVe for LSTM and contextualized representation in BERT to get word embeddings.

4.2.2. DP Training (Stage 2)

The raw user data often contain private information such as personal attributes and query logs. Thus, our foremost requirement is that the service provider only allows working with privatized input at both training and inference time, without any access to the raw data. During collaborative training, each user applies the DP mechanism to transform raw data into LDP-guaranteed latent representations before sending them to the server for task adaptation training. The crucial procedure is calibrating noise for achieving a certain privacy protection level while maintaining an acceptable level of performance in downstream tasks. Therefore, we propose a non-parametric DP layer after the encoder to achieve an LDP guarantee with detailed operations. The noise calibration algorithm depending on DP composition and sub-sampling DP amplification are described in Sec. 4.3.3.

4.2.3. DP Inference (Stage 3)

After obtaining trained privacy-preserving models, the models are released by the server and provide services to the users. The raw data which may contain sensitive information (e.g. medical history) queried by the users is converted by the local DP-layer into DP-guaranteed latent representations before being sent to the server for the inference process. The calibrated noise power depending on the up-sampling DP amplification algorithm is described in Sec. 4.3.4.

4.2.4. Inference Results Return (Stage 4)

The final layer of the neural model will decide what is the best results to respond to users' DP-protected queries according to the specific NLP tasks. Normally, the decision depends on the category scores for text classification, ranking scores for text retrieval, decode scores for text generation and so on.

4.3. Methodology

4.3.1. Non-Parametric DP-Layer

As we explained earlier in Sec. 2.3, two important operations are required to achieve a differentially private algorithm: (1) sensitivity bound; (2) noise calibration. The sensitivity bound operation is used to limit the output value of the encoder into a range as ℓ_2-norm. The noise calibration is calculated under a certain accounting framework with its DP mechanism and accounts for the required noise power for one training step protection. However, we should note that the DP guarantee for one training step does not mean its calibrated protection level ϵ referring to covers the whole training procedure. Only after compositing all training steps together by DP composition can reflect the true protection level ϵ, otherwise, it would be wrong.

To estimate the sensitivity of an arbitrary output representation from the neural layer is challenging and most previous works lack this crucial procedure [**30, 62, 7, 44, 38**] or are wrong i.e. estimate sensitivity at a wrong granularity level for achieving it, which may incur inaccurate calibration on privacy guarantee. We propose a non-parametric DP-layer by injecting calibrated noise into the clipped output of the encoder following the gradient clipping in DP-SGD [**60, 5**]. We explain this operation below.

Clipping Operation: One method to stabilize the output is clipping. Let x and x' be arbitrary inputs from the training or inference sets, and define $f(x)$ as the corresponding output of the encoder in Fig. 2(b) and Fig. 2(c). The sensitivity Δ of input $f(x)$, which is the greatest variation output for a sequence with the ℓ_2-norm is given by

$$\Delta = \max_{x,x'} ||f(x) - f(x')||_2. \tag{4.3.1}$$

Because of the randomness of training data, computing Δ is difficult. We limit f's output range by clipping each latent representation from the output of the local encoder with

$$\mathrm{CL}\left(f(x); C\right) = f(x) \cdot \min\left(1, \frac{C}{\|f(x)\|_2}\right). \tag{4.3.2}$$

where the C is a pre-defined hyper-parameter equalling to the value of sensitivity Δ.

This clipping ensures that if $\|f(x)\|_2 \leq C$, then the output value of the encoder $f(x)$ is preserved, whereas if $\|f(x)\|_2 > C$, it gets scaled down to be of norm C equalling the value of sensitivity Δ. The privacy-utility trade-off is sensitive to the quantity C because the lower the value of C, the less calibrated noise power is required for a given DP protection level. However, cutting too much of the latent representation to achieve a small C will harm the semantic features and result in a significant performance drop. Therefore, the hyper-parameter C needs to be carefully tuned.

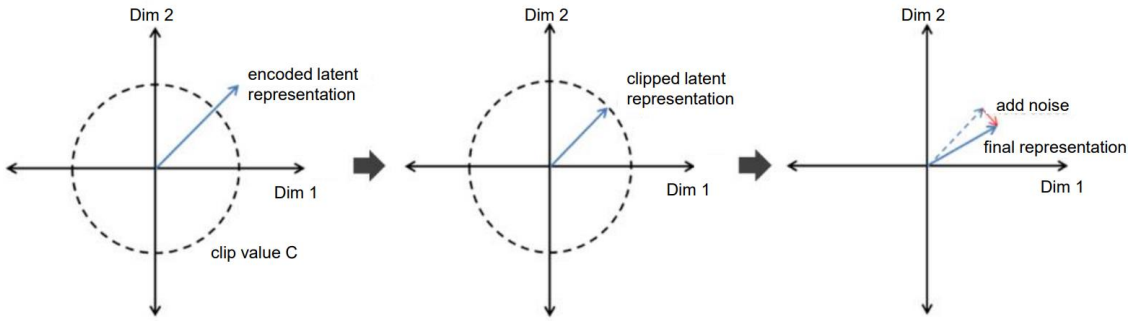


Fig. 4.3. An intuitive illustration for the text encoding, representation clipping, and noise-injecting operations within the DP-layer.

Algorithm 2 Non-parametric DP-Layer

Require: Latent representation $f(x) \in \mathbb{R}^d$, clipping value C, noise variance σ^2

1: Gaussian Mechanism: $\widetilde{x} \leftarrow \mathrm{CL}\left(f(x); C\right) + z$ with $z \sim \mathcal{N}(0, \sigma^2 I_d)$.

2: **return** $\widetilde{x}$.

As we mentioned earlier in Sec. 2.3.1, to leverage DP mechanisms for achieving DP guarantee, the widely used Laplace mechanism provides ϵ-DP with $\delta = 0$ by adding noise that follows the Laplace distribution, while the Gaussian mechanism provides (ϵ,δ)-DP, which by introducing very small δ trades off some ϵ to reduce the amount of noise required. Based on the Gaussian mechanism and after clipping, we apply additive Gaussian noise to the latent representation for privatization via the DP-layer by

$$\mathcal{M}(x, f(\cdot), \sigma) = \mathrm{CL}(f(x)) + \mathcal{N}(0, \sigma^2). \tag{4.3.3}$$

For better understanding, we provide an intuitive illustration for the text encoding, representation clipping, and noise-injecting operations within the DP-layer in Fig. 4.3 in a two-dimensional perspective. In addition, the details are also formally stated in Algorithm 2.

The advanced μ-GDP accounting method can improve the accuracy of the model while still providing a DP guarantee. The crucial point for calibrating privacy protection level by μ-GDP is to form the duality with (ϵ,δ)-DP, where the calibrated noise variance σ^2 and the DP protection level (ϵ, δ) follows Eq. 4.3.4 [**20**] and we can use it for computation.

$$\delta(\epsilon; \mu) = \Phi\left(-\frac{\epsilon}{\mu} + \frac{\mu}{2}\right) - \mathrm{e}^{\epsilon}\Phi\left(-\frac{\epsilon}{\mu} - \frac{\mu}{2}\right) \tag{4.3.4}$$

with

$$\mu = \Delta/\sigma, \tag{4.3.5}$$

where $\Phi(.)$ is the cumulative distribution function of the standard normal distribution. ϵ and δ denotes the theoretical privacy protection level, and the μ plays the role of duality parameter which determined by sensitivity Δ and noise variance σ^2.

Finally, we form a DP-layer containing clipping operation to bound the output sensitivity and additive Gaussian noise on the latent representation. A formal statement for the privacy guarantees of DP-layer Algorithm 2 is provided in Lemma 6. Thanks to the DP post-processing property, the DP-layer ensures the DP protection for the algorithm for training and inference stages containing DP composition and DP amplification. In the following, we will introduce how to account for privacy costs from each step to the whole procedure and apply DP amplification.

Lemma 6. *(DP-Layer Privacy) Let $f(x)$ be the encoder output with ℓ_2 sensitivity C given by Eq. 4.3.2. For any $\epsilon > 0$ and $\delta \in (0, 1)$, the mechanism described in Algorithm 2 is (ϵ, δ)-DP for each time of using the DP-layer.*

4.3.2. Sentence-Level DP Composition and Amplification

We first explain why achieving sentence-level DP protection is important for composition and amplification. The data granularity varies in different fields. For example, in the computer vision (CV) field, normally a picture is the smallest unit of a training sample which is composed of multiple pixels, while in NLP a training sample should commonly be a sequence that is composed of several tokens. Thus, the granularity of tokens for a sequence is corresponding to that of pixels for a picture. However, in terms of DP training, previous works [**80, 83**] performing on pictures in CV field with DP amplification techniques cannot directly apply to most NLP downstream tasks whose single input unit is a sequence. The above discussion indicates that to achieve DP amplification, we need to apply for DP protection on the whole training sample i.e. picture in the CV and the sentence in NLP. However, some existing works in NLP [**62, 81, 30**] only focus on the token-level protection but not the whole sequence. That might explain why they do not apply DP amplification.

The necessity of our implementation of sentence-level DP can be viewed from two aspects. On one hand, because the DP composition counts on how many times the DP mechanism is applied, achieving DP granularity on sentence-level rather than word-level can reduce privacy budget ϵ as shown in Fig. 4.4. Its results confirm that sentence-level DP achieves a better protection level (smaller ϵ) compared with word-level on the different numbers of words. On the other hand, it is difficult to correctly compose DP cost for a series of training steps and finally achieve a relatively small ϵ at the same time. High DP cost ϵ will lead to an extremely weak privacy protection level, which deviates from our original intention. One solution is using the model's inherent randomness such as sub-sampling and up-sampling to perform DP amplification to reduce the required noise while keeping the same privacy protection level. The data sampling should perform directly on the sample level as pictures in CV and sequences in NLP. Therefore, we use the [CLS] output of the PLMs as the latent sentence

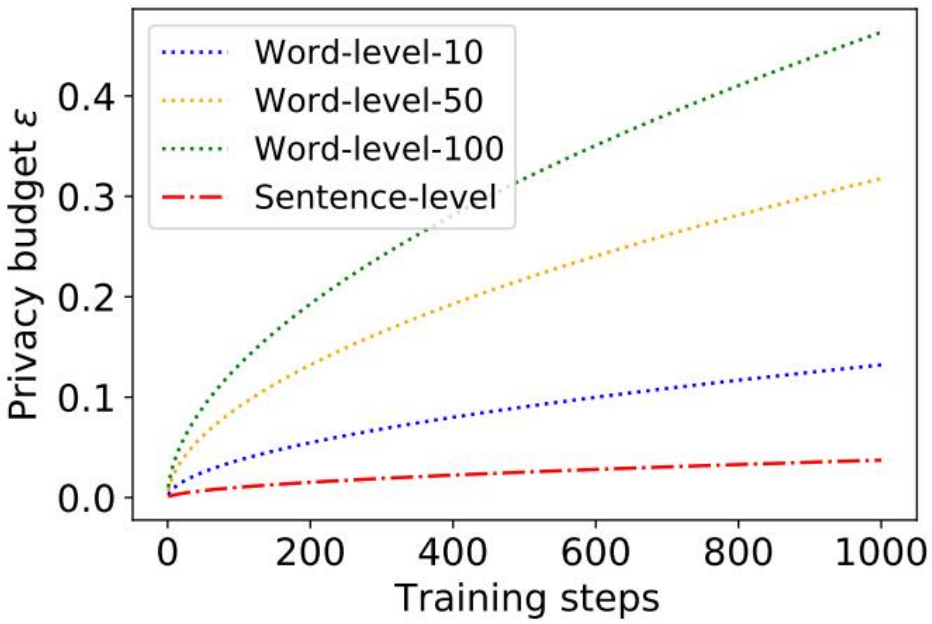


Fig. 4.4. Comparison of privacy budget ϵ between composing sentence-level training sample and word-level with the different number of tokens.

representation to achieve sentence-level protection with correct DP composition and leverage DP amplification.

In summary, correctly performing DP composition is a guarantee to provide DP protection. In experiments in Sec. 4.4.2, we will show the effect of the true and wrong cases of DP composition on the model accuracy and demonstrate the importance of DP amplification. In the following, we show how to calibrate the noise by leveraging sub-sampling and up-sampling to conduct DP amplification in the training and inference stages, respectively.

4.3.3. Gaussian-based DP Training

In previous sections, we evaluate the privacy cost for each step of input forward through DP-layer in Algorithm 2 based on μ-GDP, which measures the privacy protection level (ϵ, δ) in terms of duality parameter μ via Eq. 4.3.4 and Eq. 4.3.5. As we mentioned earlier, the computation of privacy degradation as the number of steps increases is DP composition. Now, we try to composite the privacy cost of each step to the whole training procedure.

The DP composition for each step enjoys a simple and convenient formulation in μ-GDP, for example, the n-fold composition of u_i-GDP mechanisms is $G_{\mu_1} \otimes G_{\mu_2} \otimes \cdots \otimes G_{\mu_n} = G_{\mu}$-DP with $\mu_i = \sqrt{\mu_1^2 + \cdots + \mu_n^2}$. Let x^t denote the sampled subset of data for the t-th update step (training or inference) with $|x^t|$ the number of samples and x_k^t denote the k-th sample. The neural network output of x_k^t after the DP-layer is $\frac{\Delta}{\sigma_t}$-GDP according to Eq. 4.3.5, where the duality parameter for conversion between μ-GDP and ϵ-DP is $\mu = \frac{\Delta}{\sigma_t}$. By calibrating the dynamic noise power σ_t for each step, we have the μ-GDP composition result of all the

sampled data in each step by Eq. 4.3.6.

$$\mu_t = \left|x^t\right| \frac{\Delta}{\sigma_t}. \tag{4.3.6}$$

During neural network DP training, each update step is performed on a sub-sampled training sample, which is obtained through an independent Bernoulli trial of all data samples with probability p_{train}. The dual function of Eq. 4.3.4 for each sub-sample with DP amplification can be expressed by $p_{\text{train}} \cdot G_{\mu_t} + (1 - p_{\text{train}})\text{Id}$[1] [**20**], with μ_{t} computed by Eq. 4.3.6. Usually the sub-sampling rate p_{train} is much smaller than 1, and thus the trade-off function in μ-GDP is much smaller than G_{μ_t}. In the μ-GDP framework, the DP composition and amplification occur in each step. Consider a series of T adaptive compositions of each step with μ_t, according to the recent Central Limit Theorem (CLT) for μ-GDP [**14**], the approach to $G_{\mu_{\text{tot}}}$-DP on each step duality parameter μ_t with DP amplification is given by Eq. 4.3.7. Then the DP composition from one step to all steps privacy cost is computed by Eq. 4.3.8, where the μ_{train} is the final duality parameter for privacy protection level calibration.

$$\mu_{\text{tot}} = \sqrt{\ln\left(\frac{\mu_{\text{t}}^2}{p_{\text{train}}^2 t} + 1\right)} \tag{4.3.7}$$

$$\mu_{\text{train}} = \sqrt{\mu_1^2 + \cdots + \mu_T^2} \qquad tot \in [1, T] \tag{4.3.8}$$

In practice, to perform the Gaussian-based DP training, we first set up a privacy budget parameter (ϵ, δ) which refers to the final privacy level, as well as the pre-defined sensitivity Δ and noise variance σ^2. The consumption of the privacy budget of each step is computed by the duality Eq. 4.3.6 and Eq. 4.3.7 with DP composition in batch samples and DP amplification. Then the all-steps DP composition is made based on Eq. 4.3.8. The quantity μ_{train} indicates the final duality parameter after μ-GDP accounting. The sample probability p_{train} decides the speed of the privacy budget consumption. The model training will stop at step T when the pre-defined privacy budget parameter (ϵ, δ) is achieved. Therefore, for $t \in [1, T]$, all μ_t achieves the DP composition. We estimate the overall privacy cost within each step update and then calibrate the noise power for the whole end-to-end method. The process is described in Algorithm 3. All privacy guarantee that we report keeps track of the entire μ-GDP function to find the numerical solution for ϵ given δ via a binary search.

4.3.4. Up-Sampling DP Amplification

After the privacy-preserve trained model is obtained, the same DP mechanism can be applied to the inference stage. This process can improve robustness and achieve protection

[1]Id refers to indicator function.

Algorithm 3 Gaussian-based DP Training

Require: DP budget (ϵ,δ), sensitivity Δ, noise variance σ^2, sampling rate p_{train}, encoder f, user data $\mathcal{S}(x_n)$, DP-Layer g

Initial DP budget cost $\epsilon_0 = 0$ and $t = 0$
while $\epsilon_t \leq (\epsilon,\delta)$ **do**
  Sub-sample $x \subseteq \mathcal{S}(x_n)$ with probability p_{train}
  Feature extraction: $f(x) \leftarrow x$
  DP-Layer perturbation: $\widetilde{x} \leftarrow g(f(x))$
  Compute μ_{t} by Eq. 4.3.6
  Compute μ_{train} by Eq 4.3.7
  Compute μ_{tot} by Eq. 4.3.8
  Compute DP budget cost ϵ'_t by Eq. 4.3.4
  Update model $\mathcal{F}_t$ parameters θ_t
  Update $\epsilon_{t+1} \leftarrow \epsilon_t + \epsilon'_t$, $t \leftarrow t + 1$
end while
return Trained model $\mathcal{F}$

Algorithm 4 Up-Sampling Differential Private Amplification (USDPA)

Require: Inference/query DP budget (ϵ,δ), user queries $\mathcal{Q}$, fictitious data $\mathcal{M}$, true data rate λ, sampling rate p_{query}

Mix the true query data with fictitious data by keeping the true data rate λ
Initial queried set $D = \varnothing$
Initial sampled time $t = 0$
Compute sampling rate $p_{\text{query}} = \lambda \cdot \frac{|\mathcal{M}|}{|\mathcal{M}|+|\mathcal{Q}|}$
while $|D| \leq |\mathcal{Q}|$ **do**
  Sampling query data q from the mixed dataset
  if $q \in \mathcal{Q}$ and $q \notin D$ **then**
    $D \leftarrow D \vee q$
  end if
  $t \leftarrow t + 1$
end while
Compute μ_t by Eq. 4.3.9
Compute σ by Eq. 4.3.6
return Noise power σ, sampling times t

on user queries. Although some of the current works consider inference privacy [**44, 62, 81**], none of them explore the up-sampling DP amplification on this phrase. To improve the model utility, we propose a DP amplification algorithm via up-sampling for the DP inference stage. The general idea is to introduce uncertainty into the inference data set by up-sampling it with fictitious data. We generate some fictitious samples that do not contain any private information and mix them with the true queries before randomly sampling the queries to send to the server via the DP-layer. All the true queries will be sent out via multi-up-samplings. Then, the adversaries are hard to distinguish which are the goal queries that need to be

attacked. The effectiveness of this kind of up-sampling amplification can be viewed on two sides. On one hand, the mixed fictitious samples via sampling can achieve a stronger privacy level. On the other hand, to achieve the same privacy level, the required noise in DP-layer will be significantly reduced, thus resulting in higher accuracy of downstream tasks without jeopardizing privacy.

Let Q and M denote the original and fictitious inference sets, respectively. Then we have the true data ratio $\lambda = |Q|/(|Q| + |M|)$ for the mixed data set. Originally, for each step t, we sample each user query by independent Bernoulli trial with probability p_{query}. After constructing mixed data set, the sampling probability of each true query is given by $\lambda \cdot p_{\text{query}}$. Following a similar analysis in the previous Gaussian DP training amplification, the duality parameter of the up-sampling DP amplification algorithm for the DP inference stage is

$$\mu_{\text{query}} = \lambda \cdot \sqrt{\ln\left(\frac{\mu_t^2}{p_{\text{query}}^2 t} + 1\right)} \tag{4.3.9}$$

Our up-sampling DP amplification (USDPA) mechanism is formally described in Algorithm 4. Note that the DP amplification in the inference stage does not come for free. Similar to how sub-sampling reduces the training convergence rate, up-sampling increases the query/inference times since the mixed fictitious data and uncertainty from the sampling. We give an analysis about this at Sec. 4.4.4.

4.4. Experiments

4.4.1. Experimental Setup

To evaluate our privacy-preserving GauDP model more comprehensively, we run experiments on six more datasets with different types of text classification tasks in addition to the two datasets used in Chapter 3. The information of all datasets is summarized in Table 4.1.

Dataset	Task Type	Classes	Training Sample	Test Sample
SST-2	Sentiment Analysis	2	67k	872
QNLI	Question Answering	2	104k	5.4k
QQP	Semantic Matching	2	384k	40k
MNLI	Natural Language Inference	3	241k	20k
IMDB	Sentiment Analysis	2	25k	25k
AGnews	News Categorization	4	120k	7.6k
DBpedia	Topic Analysis	9	92k	60k
SNLI	Natural Language Inference	3	511k	9.8k

Table 4.1. Statistic of datasets

We use two-layer stacked Bi-LSTM and BERT[2] from Transformer library [**74**] to perform privacy-preserving downstream tasks. For all models, we set the max sequence length as 128, the batch size as 32, the learning rate as 2e-5, and the dropout rate as 0.1. For the Bi-LSTM model, we set the input embedding dimension as 200 and the hidden layer size as 256. For BERT, we keep all the other hyper-parameters as the original pre-training setting.

4.4.2. Re-examining DP Composition and Amplification

Some previous works [**44, 62, 30, 38, 35, 7**] to produce differentially private text representation seems to perform well on the privacy-utility trade-off. However, the privacy protection is not as strong as they claim, because of the lack of DP composition and sensitivity estimation or wrong on them. For example, they [**44, 38, 35**] only calibrate one step DP protection level ϵ rather than compositing all training steps together, or they [**44, 38**] estimate sensitivity at a wrong granularity level (e.g. need to calculate on a sample but only on a semantic feature coordinate within a sample) or even lack of sensitivity estimation [**62, 30, 7**]. Some proof is shown in the other works [**34**]. The wrong methods for DP composition and sensitivity estimation will lead to inaccurate privacy protection level calibration. Besides, large privacy cost ϵ reported in [**35, 62**] implies an extremely weak privacy guarantee. In this section, we re-examine the role of DP composition and DP amplification according to the performance gap among three different settings. Meanwhile, we compare the different DP accounting methods under the same sensitivity estimation as Eq. 4.3.1.

Experiment On Different DP Composition and Amplification. The absence of DP composition will result in the inaccurate final privacy protection level, because it only counts the privacy cost of one-step of training to produce the DP-protective representations without all-steps composition. The true total privacy cost would scale to $\epsilon = \mathcal{O}(T)$ based on the basic composition or $\epsilon = \mathcal{O}(\sqrt{T})$ based on the advanced composition [**26**]. We conduct experiments with three different settings on SST-2 and IMDB datasets: (1) Compose privacy cost ϵ on all training steps with corresponding calibrated noise **without** DP amplification. (2) Compose privacy cost ϵ on all training steps with corresponding calibrated noise **with** DP amplification. (3) Account privacy cost ϵ on only one training step with corresponding calibrated noise with DP amplification and we should note that this is a **inaccurate/wrong** case as reported in some previous works because it only counts one-step privacy cost without DP composition. We use three different PLMs including Bert, Albert, and Distilbert as the encoder to explore the impact of different models on DP training. All the privacy protection level calibrations are based on μ-GDP.

The experiment results are shown in Table 4.2. The all-steps composition without DP amplification performs much worse than those with amplification techniques. Though the

[2]including its variants such as Albert, Distilbert, and Roberta for some experiments.

Models	ϵ	Accuracy on SST-2			Accuracy on IMDB		
		all-steps w/o amplification	all-steps **w/ amplification**	one-step **Inaccurate**	all-steps w/o amplification	all-steps **w/ amplification**	one-step **Inaccurate**
Bert	1	59.40	82.45	90.37	58.79	79.24	89.38
	4	74.77	88.88	90.14	76.22	87.42	88.04
	8	86.12	90.60	90.83	86.77	88.42	88.90
Albert	1	57.57	80.39	86.58	57.94	78.89	88.83
	4	72.82	81.19	84.86	70.35	86.15	88.66
	8	82.91	86.24	83.60	84.95	87.19	88.40
Distilbert	1	59.40	81.54	88.19	58.15	78.92	89.02
	4	75.57	87.16	88.76	76.18	86.15	88.46
	8	84.75	89.68	90.02	84.94	88.10	88.95

Table 4.2. Accuracy of re-exam on three DP composition and DP amplification settings.

one-step DP composition achieves relatively high accuracy and it seems that privacy implementation has no negative impact on the utility, the ϵ of them only accounts for one training step privacy cost rather than the whole training procedure and thus cannot reflect the true protection level. The correct way is to composite all training steps together by DP composition as shown by all-steps, where we can find a significant performance drop. Thus, using DP amplification is necessary which can greatly improve the performance of the privacy-preserving model, especially under the high privacy level (small ϵ). We can also find that using Bert as an encoder is better than Albert and Distilbert, which might indicate a larger PLM is more suitable to be an encoder for differentially private training. Further, in Fig. 4.5, we show the noise power (variance) required to achieve the corresponding privacy level ϵ among these three settings with 100k size data samples. To achieve stronger privacy protection (small ϵ), the all-data composition without amplification requires several times more noise than those with amplification technique and wrong without all-data composition. A large amount of noise would dramatically hurt the original semantics and result in a significant performance drop. Both results demonstrate the effectiveness of DP amplification and indicate that we need to carefully implement correct DP composition in all training steps. The all-steps DP composition avoids taking the privacy protection calibration result of one training step as the whole training procedure result, which is a common mistake in some previous works.

Comparison of DP Accounting Method. To choose the suitable privacy accounting method for DP experiments, we compare three different privacy accounting methods, including Rényi DP (RDP) [**53, 8**], μ-Gaussian DP with CLT (GDP+CLT) [**20**] and Compose Tradeoff Function [**33**], under the same sensitivity estimation with 100k steps DP composition and corresponding DP amplification. The results are shown in Fig. 4.6 and we can see that to achieve the same privacy level ϵ, RDP requires more noise than the other two. The Compose Tradeoff Function is a good approximation when the privacy level is high (small ϵ) but is still worse than the GDP+CLT method in some cases. Therefore, we use

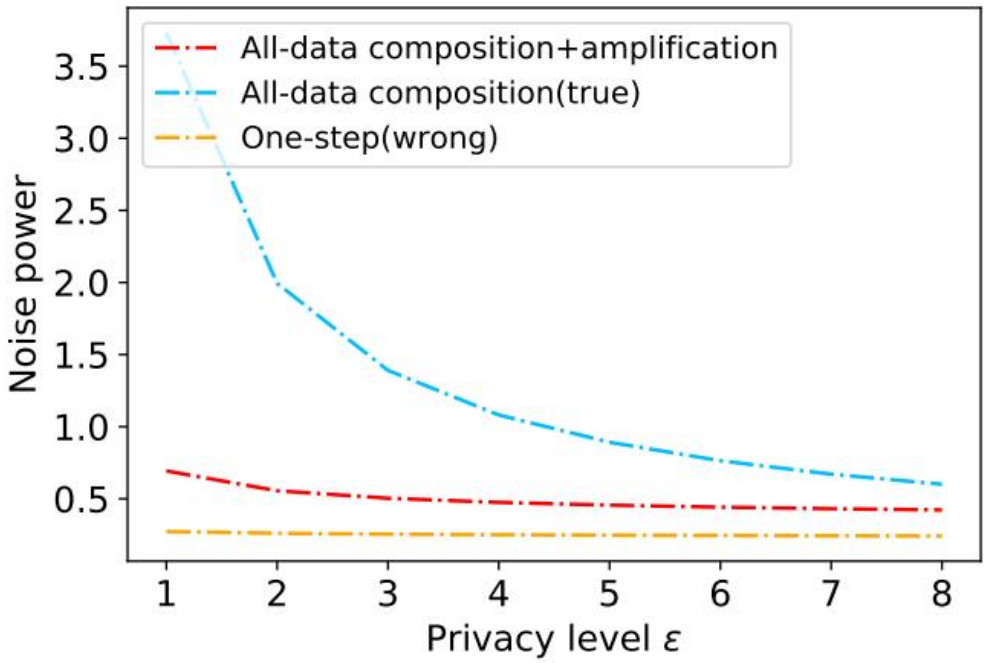


Fig. 4.5. The relation of privacy level and noise among three different settings.

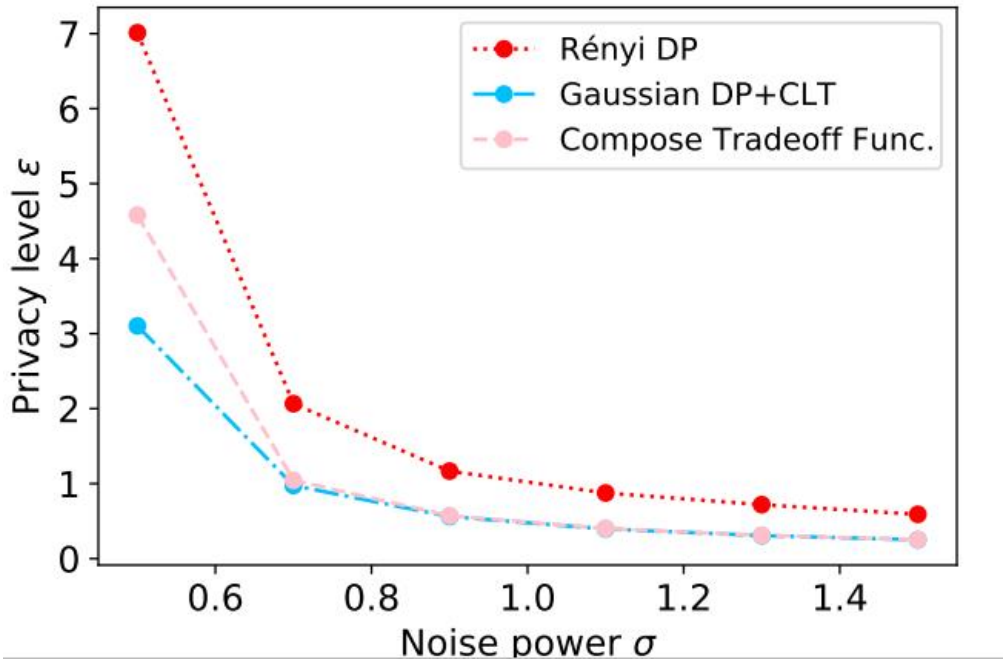


Fig. 4.6. The comparison of privacy level and required noise among three methods.

the GDP+CLT method under the μ-GDP framework for all our following experiments to calibrate noise.

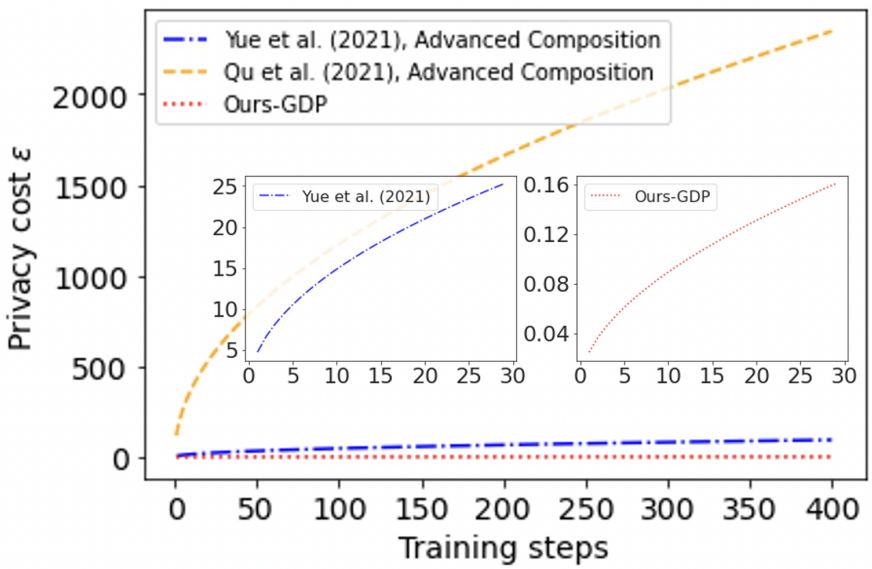


Fig. 4.7. The comparison of DP composition privacy cost via various methodologies.

4.4.3. Privacy-Accuracy Trade-off on Training and Inference

The ultimate goal is to achieve a reasonable total privacy cost for the entire training and inference separately. The total DP cost ϵ, which is a function of training steps, boosts using the advanced composition method [**26**] compared with our Gaussian private training method, as shown in Fig. 4.7. The greatest saving of privacy cost of our method is due to the proposed DP layer benefits from the model's inherent randomness, i.e. sub-sampling for training and a tight DP composition. Therefore, we can calibrate the noise power tightly as Algorithm 2 so as to achieve a better privacy-accuracy trade-off.

Baseline. To protect text privacy at the token-level, a relaxation of the standard DP definition known as d_χ-privacy [**17**] and the corresponding mechanisms [**30, 81, 62**] have recently been proposed. Because of the unique mechanism used, sampling amplification and tight composition are still absent. As a result, only the DP cost of each training step can be calculated in previous methods [**30, 81, 62**]. To account for the total privacy cost for all data used at sequence-level protection, as far as we know, the best way for them is to apply the advanced composition [**26**] as Eq. 2.3.3 to achieve an overall privacy cost.

We provide the advanced composition of existing DP text protection methods using d_χ-privacy as well as the utility of the null privacy case, which serves as the upper bound.

Training Phrase. We first compare the performance of our proposed method with previous works [**30, 81, 62**] at different DP cost constraints for the entire training datasets. Since none of the existing methods considers the privacy protection of inference data, we consider this case separately in the next paragraph. To make a fair comparison, we select the smallest d for neighbor search as a lower bound from Fig. 3 in paper [**62**], which gives the strongest

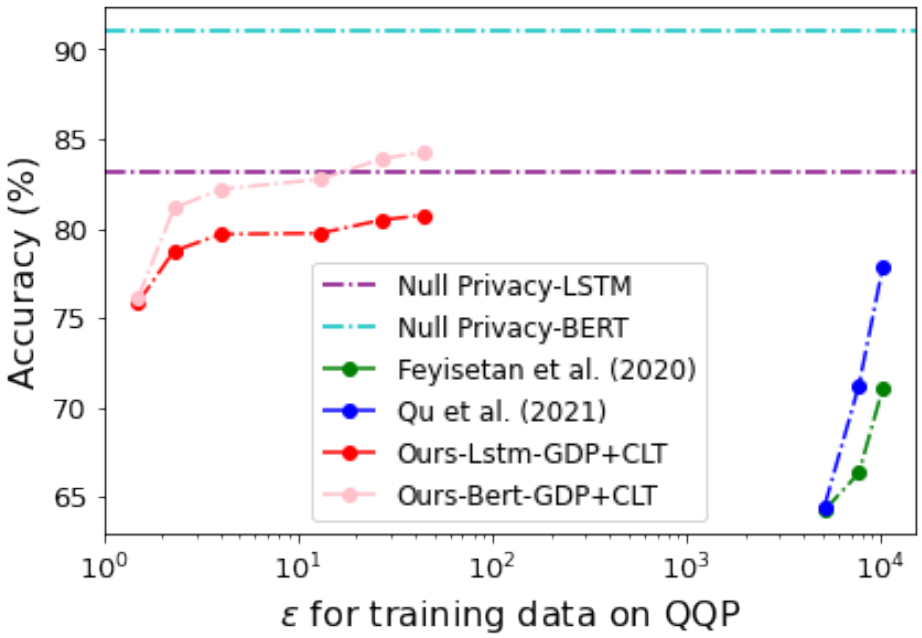


Fig. 4.8. Accuracy vs. training privacy on QQP.

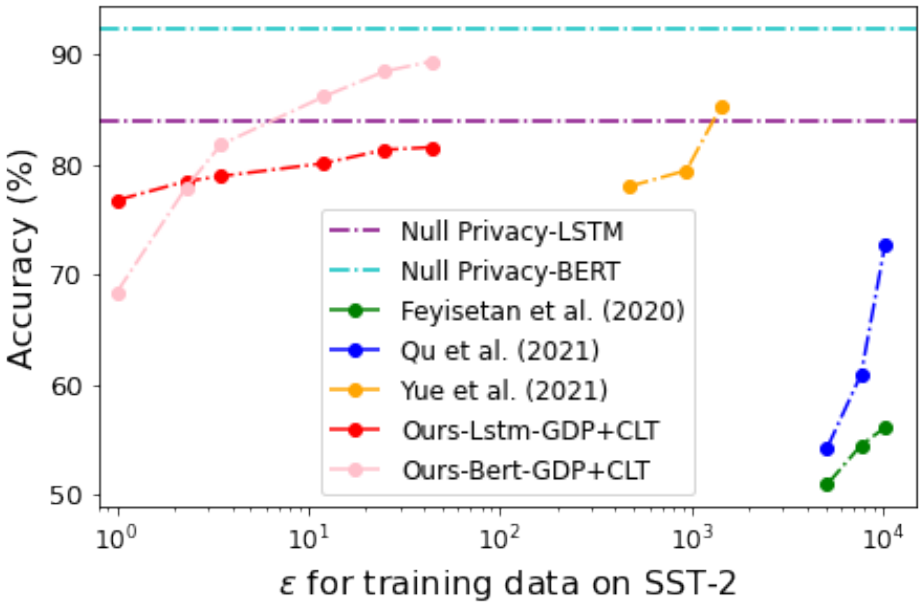


Fig. 4.9. Accuracy vs. training privacy on SST-2.

privacy. For both QQP and SST-2 datasets, it is shown consistently in Fig. 4.8 and Fig. 4.9 that missing a tight DP accounting method results in the total privacy parameter ϵ scaling to more than 5000 in the existing works, which does not guarantee any privacy for the whole dataset at sequence-level even though they provide reasonable privacy protection for each step. Moreover, due to the random noise applied to the embedding layer, the performance of these methods [**30, 62**] degrades significantly compared to the null privacy case. These

Inference ϵ	SST-2 Inference Accuracy			QQP Inference Accuracy		
	no USDPA	**+USDPA**	**+USDPA (Retrain)**	no USDPA	**+USDPA**	**+USDPA (Retrain)**
0.3	76.72	**+1.25**	**+3.31**	75.83	**+1.17**	**+3.82**
0.6	78.44	**+0.77**	**+1.72**	78.72	**+0.53**	**+1.29**
1	78.90	**+0.80**	**+1.25**	79.67	**+0.11**	**+0.64**
3	80.70	-0.19	**+0.77**	79.82	**+0.10**	**+0.69**
6	81.30	-0.46	**+0.69**	80.47	-0.29	**+0.66**
9	81.53	-1.60	**+0.14**	80.74	-0.15	**+0.47**

Table 4.3. Accuracy improvement by USDPA algorithm with different setting at various privacy level for inference data.

models even tend to become a random classifier when $\epsilon = 5100$, but still with weak privacy protection. By contrast, our proposed model with Algorithm 3 improves the performance, which approaches the null privacy case for the SST-2 dataset for both the LSTM and BERT encoders. Moreover, it is observed that the performance loss to the non-DP version of the BERT model is larger than that of the LSTM model because its large representation dimension is more sensitive to clipping and noise.

Inference Phrase. We further test the proposed up-sampling DP amplification algorithm (USDPA) for accuracy improvement. First, we directly apply the USDPA Algorithm 4 on inference/query data to check the improvement based on the model obtained by the training phrase which provides protection for training data. The results are shown in Table 4.3, where we can find the improvement when $\epsilon \leq 1$, but degradation on large $\epsilon \geq 3$. Though such a model can protect both training and inference data now, in this case, the noise power for inference is not consistent with the training case. For example, the amplification effect depends on the size of the sampled set. If the size gap between the training set and the test set is large, the injected noise amount for the latent representation of the training and inference stage will be significantly different, which in turn hurts the model robustness. The change of ϵ would lead to the same problem of misalignment of noise amount, as the privacy level is not linear to the change of noise amount as shown in Fig. 4.6. Therefore, when ϵ is relatively large (e.g. $\epsilon > 3$), the training stage requires more noise than inference. The noise amount calibrated by the training set for model training but applied to the inference set might lead over-protective problem and thus results in the degradation of accuracy. As we mainly aim at protecting inference data here, a common practice [**44**] is to retrain the model with the noise amount calibrated by the size of the inference set and then apply the USDPA algorithm for DP amplification. Though the noise for the training phrase is calibrated by inference set within the retrained model, it can still provide protection for training data to some extent. The above results show that the up-sampling technique improves accuracy, and the stronger the privacy guaranteed, the larger an accuracy gain is obtained. In practice, the retrain procedure is optional according to the dataset size, and we can directly apply USDPA to inference phrases in most cases.

Method	MNLI		QQP		QNLI		SST-2		Avg.	
Yu et al. (2021) [**79**]	-	78.6	-	84.8	-	**86.2**	-	**91.5**	-	85.28
Li et al. (2021) [**41**]	**82.29**	**83.22**	85.41	86.15	84.62	84.81	86.12	85.89	**84.61**	85.02
Ours-Private-Bert	76.80	78.65	**86.40**	**86.95**	**84.88**	85.77	**87.50**	89.79	83.90	**85.29**
Ours-Private-RoBerta	79.36	81.65	**85.49**	**86.87**	84.09	**86.00**	82.91	87.84	82.96	**85.59**
No Privacy Bert	84.52		90.65		90.61		92.31		89.52	
No Privacy Roberta	86.19		91.20		91.64		93.69		90.68	
$\epsilon =$ (RDP)	3	8	3	8	3	8	3	8	3	8
$\epsilon \approx$ (μ-GDP)	2.52	5.83	2.53	5.85	2	4.75	1.73	4.33	<2.53	<5.85

Table 4.4. The comparison with centralized training methods.

Comparison with Centralized Training. We also compare our local differentially private training model with existing start-of-the-art centralized differentially private training models [**79, 41**] under the same privacy level and model size. These centralized training methods try to achieve a better privacy-utility trade-off by fully fine-tuning PLMs with DP gradient perturbation and addressing the computational challenge of running the DP-SGD algorithm with large PLMs.

From the efficiency perspective, although the two centralized training methods exploit a re-parameterized gradient perturbation method and a memory-saving technique to improve the training efficiency, they still require 6 times more time for each epoch training compared with our GauDP model under the LDP setting. On the effectiveness side, the privacy-accuracy comparison is shown in Table 4.4. For fair analysis, we implement our Gaussian-based DP training under two privacy levels ϵ accounted by RDP and μ-GDP with Bert and Roberta following Yu et al. [**79**] and Li et al. [**41**], respectively. We can find that our method performs better on three datasets except for MNLI, especially under the strong privacy protection level. This might be because the semantic change by the privacy-preserving operation is more sensitive for the natural language inference prediction in MNLI. Besides, the Roberta encoder performs well in two natural language inference tasks, while the Bert encoder is good at the other tasks. The results indicate that we need to choose a suitable encoder with DP protection for different tasks.

4.4.4. Detailed Analysis

The design of the entire local differential privacy-preserving NLP model makes several impacts. Here, we conduct a series of experiments to analyze these impacts, aiming to gain insight into how to achieve a better privacy-utility trade-off.

DP-Layer Implementation Position Impact. We first analyze the implementation position impact of the proposed DP layer. We apply it after the embedding layer as existing works [**30, 62, 81**] shown at Fig. 2(a) and after the encoder as other previous

Training ϵ	QQP		SST-2	
	Token Rep.	Latent Rep.	Token Rep.	Latent Rep.
0.3	71.53	75.83	68.23	76.72
0.6	72.85	78.72	71.33	78.44
1	74.25	79.67	73.32	78.90
3	74.51	79.82	73.51	80.70
6	75.51	80.47	74.20	81.30
9	75.71	80.74	74.54	81.53
Null Privacy	83.11		83.91	

Table 4.5. DP layer applied to the token representation versus that applied to the latent representation base on Bi-LSTM model within μ-GDP accounting framework.

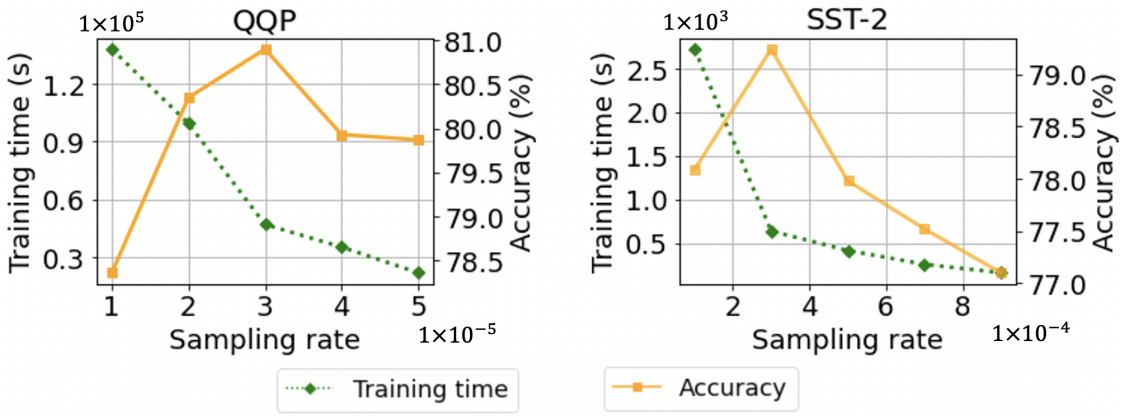


Fig. 4.10. Effectiveness and efficiency relation influenced by the sub-sampling rate for training.

works [**44, 35, 38, 52**] shown at Fig. 2(b), respectively. Table 4.5 illustrates their privacy-accuracy results. The difference between them is where to inject the noise, on token representation output by embedding layer or latent representation generated by the encoder. We want to observe the suitable position for differentially private training. Compared with applying to latent representations, being applied to the token representation is more sensitive to the random noise privatization and results in performance degradation for downstream tasks. Using the DP-layer directly on the latent representation, on the other hand, improves accuracy by 4% to 8% on both QQP and SST-2 datasets.

Sub-sampling Rate Impact. We also explore how the sub-sampling rate p_{train} affects model accuracy and training efficiency. The smaller the p_{train}, the smaller the sampled batch size, and thus the slower the convergence. However, according to Eq. 4.3.7, smaller p_{train} leads to larger DP amplification, resulting in a lower calibrated noise power. As a result, there might be a "best" p_{train} to choose from. Here, we pre-define the privacy cost $\epsilon = 6$ for SST-2 and QQP datasets to test the effect of p_{train}. The results of Fig. 4.10 agree with our intuition, and there is a p_{train} that produces the highest accuracy given a fixed privacy cost.

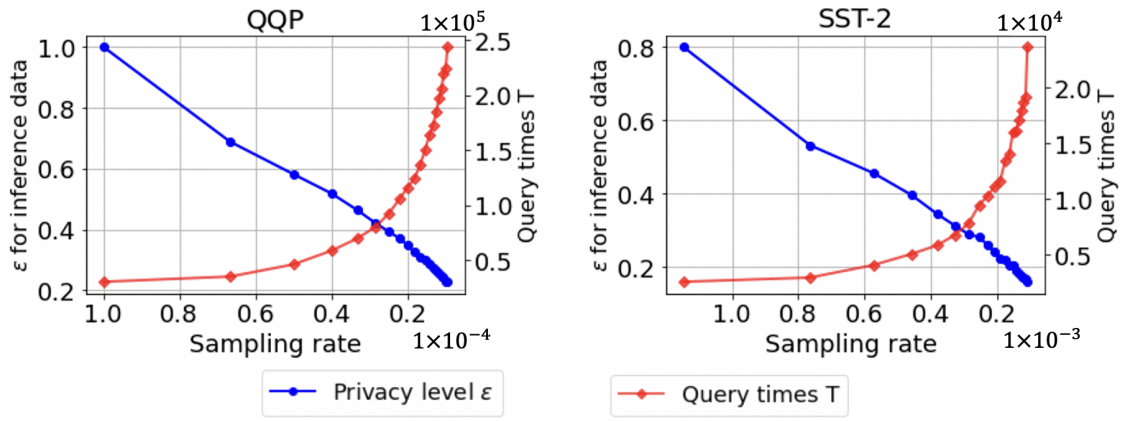


Fig. 4.11. The relation between privacy cost, sampling rate $q \cdot p_{\text{query}}$ and the query times to complete all the test samples.

Model	ϵ	IMDB	AGnews	DBpedia	SNLI	Avg.
Bert+Laplace	1	69.37	58.55	61.86	69.55	64.83
	2.5	76.41	68.07	71.08	74.92	72.62
	4	78.79	71.80	73.37	77.54	75.28
Bert+Gaussian	1	79.63	70.96	75.68	80.01	76.57
	2.5	82.68	80.25	86.73	83.81	83.37
	4	87.42	83.23	87.11	85.23	85.75
Null Privacy		89.01	93.26	99.52	89.98	92.94

Table 4.6. Accuracy of the noise type generated by different DP mechanisms on four datasets.

In practice, we can try to tune this parameter to achieve "optimal" performance based on demand.

Up-sampling Rate Impact. Similar to the above analysis, we examine the impact of $\lambda \cdot p_{\text{query}}$ in USDPA Algorithm 4 for inference. It is expected that the smaller the $\lambda \cdot p_{\text{query}}$ is, the larger the accuracy gain we can obtain from the DP amplification. However, it is also evident that we need more sampling times and inference steps to finish all the test samples, which further increases the privacy cost. Consistent with the above analysis, Fig. 4.11 illustrates the relation among sampling rate, query times, and corresponding privacy level ϵ. In practice, we can determine specific $\lambda \cdot p_{\text{query}}$ values based on query time and privacy protection level requirements. The ratio of fictitious data to true data is set to $0, 0.5, 1, 1.5, \ldots, 8.5, 9$, in our experiment as shown in Fig. 4.11, and $\lambda \cdot p_{\text{query}}$ is set to be the reciprocal of total data size. Based on the results, we can set the sampling rate according to the actual required privacy protection level and the efficiency limits.

Noise Type Impact. The types of random DP noise are generated by the various DP mechanisms. These typical random noises are sampled from a specific distribution (e.g. Laplace or Gaussian). Thus, the noise types might have a specific influence on downstream

Sensitivity C	SST-2		IMDB		AGnews		QNLI	
	σ	Acc.	σ	Acc.	σ	Acc.	σ	Acc.
1	0.75	82.34	0.94	**79.63**	0.67	71.56	0.68	**82.26**
0.9	0.67	81.65	0.85	79.24	0.61	**71.96**	0.61	82.15
0.7	0.52	**82.45**	0.66	78.52	0.47	71.61	0.48	81.12
0.5	0.37	77.87	0.47	78.67	0.34	70.47	0.34	81.12
0.3	0.22	81.42	0.28	79.09	0.20	70.00	0.21	81.22
0.1	0.07	78.10	0.09	76.26	0.07	68.72	0.07	78.52

Table 4.7. Accuracy on four datasets with different sensitivity value C and noise variance σ under $\epsilon = 1$ protection level.

tasks. Most current works [**30, 62, 38, 35**] use the Laplace mechanism to achieve DP protection because its accounting framework makes it easy to estimate the privacy level. Here, we investigate the influence of noise type by applying our method to the different downstream tasks. We fix the privacy level ϵ and calibrate the required noise power by the Laplace mechanism with RDP and Gaussian mechanism with μ-GDP to evaluate the model performance. The results are shown in Table 4.6, the Gaussian mechanism performs better than the Laplace mechanism in all cases. This might be because the Laplace distribution is sharper than the Gaussian distribution which results in the random Laplace noise having a more erratic effect on semantic changes. Thus, designing a DP training algorithm that can accurately calibrate the privacy level under the Gaussian mechanism may be the appropriate approach.

Sensitivity and Noise Power Tuning. As mentioned in Sec. 4.3.1, we need to clip the latent representation with value C to bound the sensitivity for privacy calibration. Though the smaller the parameter C, the less noise is required which can improve the model utility. However, a small clip value of C will hurt the semantic information of the latent vector and result in performance degradation. Therefore, the balance between sensitivity value and required noise power is difficult to control in practice. Here, we provide an empirical analysis of it by evaluating four popular datasets with our GauDP model using the Bert encoder.

The experiment results are shown in Table 4.7. We can see each dataset would have its most suitable sensitivity clip value C for the best accuracy. Though the required noise variance is positively related to sensitivity, the larger C is not appropriate in most cases. In practice, the DP parameter C is a hyper-parameter to be carefully tuned to balance the trade-off between utility and privacy which is similar to the learning rate. Its value determines the corresponding noise variance and model accuracy.

4.5. Conclusion

In this chapter, we study how to protect the privacy of local user data while keeping the model accuracy by designing a Gaussian-based local differentially private model *(GauDP)*. It protects the privacy of local user data at the sequence-level by producing private latent representations while keeping the model's accuracy. Specifically, we propose a DP-Layer based on the Gaussian mechanism with sensitivity bound for privacy calibration. Two algorithms for training and inference phrases via implementing DP composition and DP amplification by sampling techniques within the μ-GDP accounting framework are proposed. Extensive experiments show that the GauDP model successfully reduces calibrated noise and achieves a significant accuracy improvement while lowering total privacy costs to less than 10 for both the training and inference stages. A series of detailed analyses provide additional insights into privacy-preserving NLP and generalize future explorations.

Chapter 5

Conclusion and Future Work

Data privacy is a growing problem in modern life. The great predictive power of neural models comes with great privacy risks, which might enable user privacy leakage during deep learning training with a large-scale corpus. Despite the fact that some efforts for text protection have been made by various privacy-preserving methods, there are only a few works in differentially private text privatization, probably due to its intrinsic difficulty. Better DP protective algorithms for provable and quantifiable privacy guarantees are thus needed. This thesis attempts to study the strong differential privacy for text protection, which aims to improve the DP-NLP task performance from two perspectives: directly privatizing raw text or producing DP-protective latent representations.

The core issue of text protection is how to maintain good utility and strong privacy protection at the same time under a practical scenario. However, most previous works fail to do so, due to over-protection or inaccurate privacy calibration. In this study, we consider a local privacy setting where the data owners can choose to privatize their text locally by a certain text privatization mechanism before releasing them. We propose two methods based on producing differentially private text or differentially private latent representation to push the privacy-utility trade-off boundary. The first method is a customized differentially private text privatization mechanism named *CusText* that provides adaptive privacy protection at the token-level. The CusText integrates a novel sampling function by designing a suitable score function on top of the Exponential mechanism and providing each input token its own customized output set to boost the utility of privatized text. We also propose two new text privatization strategies to improve the utility of privatized text without compromising privacy. The second method is a Gaussian-based local differentially private model named *GauDP* that protects the privacy of local user data at the sequence-level by private latent representations while keeping the model's accuracy. The GauDP model includes a non-parametric DP-layer applied to the latent representation on the user side, DP amplifications for training/inference data via sub-sampling/up-sampling, tight DP composition, and noise

calibration algorithms for privacy accounting based on Gaussian mechanism and μ-GDP framework.

To understand the function of each component of the methodology, our experiments are conducted on several text classification datasets with various models. The experimental results and the comparison with existing works show the effectiveness of our approaches on both the protection of text privacy and the utility of the protection schema. The experiments also provide more insights into privacy-preserving NLP.

Despite the good results we obtained in our experiments, there is still a long way to go to further explore the cross-area of NLP and DP. Towards better text protection, the method can be improved in several ways in the future. On one hand, to produce private text, we can design a more advanced customized mechanism by assigning a variable size of the output set for each input token and looking for a better way to identify the sensitive tokens in the text rather than based on rules. On the other hand, to produce private latent representations, we can explore a more efficient and accurate noise calibration privacy accounting algorithm as well as corresponding DP composition and DP amplification techniques. It is also critical to designing an algorithm to automatically learn privacy parameters such as clip value and noise variance. In addition, we can try to reduce communication costs for user-server training by dimensionality reduction method.

References

[1] California Consumer Privacy Act (CCPA) 2020. California Consumer Privacy Act (CCPA). https://oag.ca.gov/privacy/ccpa. (2020). Accessed: 2020-02-14.

[2] Apple is using differential privacy to help discover the usage patterns of a large number of users without compromising individual privacy.

[3] Federated learning: Collaborative machine learning without centralized training data.

[4] Uber security (2017). uber releases open source project for differential privacy.

[5] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar et Li Zhang : Deep learning with differential privacy. *In Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[6] Charu C Aggarwal : On k-anonymity and the curse of dimensionality. *In VLDB*, volume 5, pages 901–909, 2005.

[7] Rohan Anil, Badih Ghazi, Vineet Gupta, Ravi Kumar et Pasin Manurangsi : Large-scale differentially private bert. *arXiv preprint arXiv:2108.01624*, 2021.

[8] Borja Balle, Gilles Barthe et Marco Gaboardi : Privacy amplification by subsampling: Tight analyses via couplings and divergences. *arXiv preprint arXiv:1807.01647*, 2018.

[9] Priyam Basu, Tiasa Singha Roy, Rakshit Naidu, Zumrut Muftuoglu, Sahib Singh et Fatemehsadat Mireshghallah : Benchmarking differential privacy and federated learning for bert models. *arXiv preprint arXiv:2106.13973*, 2021.

[10] Michael Bendersky, Xuanhui Wang, Donald Metzler et Marc Najork : Learning from user interactions in personal search via attribute parameterization. *In Proceedings of the tenth ACM international conference on web search and data mining*, pages 791–799, 2017.

[11] Yoshua Bengio, Réjean Ducharme et Pascal Vincent : A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13, 2000.

[12] Abhishek Bhowmick, John Duchi, Julien Freudiger, Gaurav Kapoor et Ryan Rogers : Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*, 2018.

[13] Justin Brickell et Vitaly Shmatikov : The cost of privacy: destruction of data-mining utility in anonymized data publishing. *In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 70–78, 2008.

[14] Zhiqi Bu, Jinshuo Dong, Qi Long et Weijie J Su : Deep learning with gaussian differential privacy. *Harvard data science review*, 2020(23), 2020.

[15] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos et Dawn Song : The secret sharer: Evaluating and testing unintended memorization in neural networks. *In 28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 267–284, 2019.

[16] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson *et al.* : Extracting training data from large language models. *In 30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[17] Konstantinos Chatzikokolakis, Miguel E Andrés, Nicolás Emilio Bordenabe et Catuscia Palamidessi : Broadening the scope of differential privacy using metrics. *In International Symposium on Privacy Enhancing Technologies Symposium*, pages 82–102. Springer, 2013.

[18] Maximin Coavoux, Shashi Narayan et Shay B Cohen : Privacy-preserving neural representations of text. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, 2018.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee et Kristina Toutanova : Bert: Pre-training of deep bidirectional transformers for language understanding. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[20] Jinshuo Dong, Aaron Roth et Weijie Su : Gaussian differential privacy. *Journal of the Royal Statistical Society*, 2021.

[21] John C Duchi, Michael I Jordan et Martin J Wainwright : Local privacy and statistical minimax rates. *In 2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

[22] Christophe Dupuy, Radhika Arava, Rahul Gupta et Anna Rumshisky : An efficient dp-sgd mechanism for large scale nlp models. *arXiv preprint arXiv:2107.14586*, 2021.

[23] Cynthia Dwork et Jing Lei : Differential privacy and robust statistics. *In Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 371–380, 2009.

[24] Cynthia Dwork, Frank McSherry, Kobbi Nissim et Adam Smith : Calibrating noise to sensitivity in private data analysis. *In Theory of cryptography conference*, pages 265–284. Springer, 2006.

[25] Cynthia Dwork, Aaron Roth *et al.* : The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[26] Cynthia Dwork, Guy N Rothblum et Salil Vadhan : Boosting and differential privacy. *In 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60. IEEE, 2010.

[27] David Evans, Vladimir Kolesnikov et Mike Rosulek : A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, 2(2-3), 2017.

[28] David Evans, Vladimir Kolesnikov, Mike Rosulek *et al.* : A pragmatic introduction to secure multi-party computation. *Foundations and Trends® in Privacy and Security*, 2(2-3):70–246, 2018.

[29] O. Feyisetan, T. Diethe et T. Drake : Leveraging hierarchical representations for preserving privacy and utility in text. *In 2019 IEEE International Conference on Data Mining (ICDM)*, 2019.

[30] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake et Tom Diethe : Privacy-and utility-preserving textual analysis via calibrated multivariate perturbations. *In Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 178–186, 2020.

[31] Oluwaseyi Feyisetan, Sepideh Ghanavati et Patricia Thaine : Workshop on privacy in nlp (privatenlp 2020). *In Proceedings of the 13th International Conference on Web Search and Data Mining*, pages 903–904, 2020.

[32] Ian J Goodfellow, Jonathon Shlens et Christian Szegedy : Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[33] Sivakanth Gopi, Yin Tat Lee et Lukas Wutschitz : Numerical composition of differential privacy. *arXiv preprint arXiv:2106.02848*, 2021.

[34] Ivan Habernal : When differential privacy meets nlp: The devil is in the detail. *In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1522–1528, 2021.

[35] Jack Hessel et Alexandra Schofield : How effective is bert without word ordering? implications for language understanding and data privacy. *In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 204–211, 2021.

[36] Marija Jegorova, Chaitanya Kaul, Charlie Mayor, Alison Q O'Neil, Alexander Weir, Roderick Murray-Smith et Sotirios A Tsaftaris : Survey: Leakage and privacy at inference time. *arXiv preprint arXiv:2107.01614*, 2021.

[37] Robin Jia et Percy Liang : Adversarial examples for evaluating reading comprehension systems. *In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, 2017.

[38] Satyapriya Krishna, Rahul Gupta et Christophe Dupuy : Adept: Auto-encoder based differentially private text transformation. *In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2435–2439, 2021.

[39] Cheng Li, Mingyang Zhang, Michael Bendersky, Hongbo Deng, Donald Metzler et Marc Najork : Multi-view embedding-based synonyms for email search. *In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 575–584, 2019.

[40] Ninghui Li, Tiancheng Li et Suresh Venkatasubramanian : t-closeness: Privacy beyond k-anonymity and l-diversity. *In 2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE, 2007.

[41] Xuechen Li, Florian Tramèr, Percy Liang et Tatsunori Hashimoto : Large language models can be strong differentially private learners. *arXiv preprint arXiv:2110.05679*, 2021.

[42] Yitong Li, Timothy Baldwin et Trevor Cohn : Towards robust and privacy-preserving text representations. *In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 25–30, 2018.

[43] Ming Liu, Stella Ho, Mengqi Wang, Longxiang Gao, Yuan Jin et He Zhang : Federated learning meets natural language processing: A survey. *arXiv preprint arXiv:2107.12603*, 2021.

[44] Lingjuan Lyu, Xuanli He et Yitong Li : Differentially private representation for nlp: Formal guarantee and an empirical study on privacy and fairness. *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2355–2365, 2020.

[45] Lingjuan Lyu, Han Yu et Qiang Yang : Threats to federated learning: A survey. *arXiv preprint arXiv:2003.02133*, 2020.

[46] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke et Muthuramakrishnan Venkitasubramaniam : l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3–es, 2007.

[47] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson et Blaise Aguera y Arcas : Communication-efficient learning of deep networks from decentralized data. *In Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[48] H Brendan McMahan, Daniel Ramage, Kunal Talwar et Li Zhang : Learning differentially private recurrent language models. *In International Conference on Learning Representations*, 2018.

[49] Frank McSherry et Kunal Talwar : Mechanism design via differential privacy. *In 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

[50] Frank D McSherry : Privacy integrated queries: an extensible platform for privacy-preserving data analysis. *In Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.

[51] Tomas Mikolov, Kai Chen, Greg Corrado et Jeffrey Dean : Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[52] Fatemehsadat Mireshghallah, Huseyin Inan, Marcello Hasegawa, Victor Rühle, Taylor Berg-Kirkpatrick et Robert Sim : Privacy regularization: Joint privacy-utility optimization in language-models. *In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3799–3807, 2021.

[53] Ilya Mironov : Rényi differential privacy. *In 2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[54] Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen et Steve Young : Counter-fitting word vectors to linguistic constraints. *In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, California, juin 2016. Association for Computational Linguistics.

[55] Takao Murakami et Yusuke Kawamoto : {Utility-Optimized} local differential privacy mechanisms for distribution estimation. *In 28th USENIX Security Symposium (USENIX Security 19)*, pages 1877–1894, 2019.

[56] Yiwen NIE, Wei Yang, Liusheng Huang, Xike Xie, Zhenhua Zhao et Shaowei Wang : A utility-optimized framework for personalized private histogram estimation. *IEEE Transactions on Knowledge and Data Engineering*, 31(4):655–669, 2019.

[57] Xudong Pan, Mi Zhang, Shouling Ji et Min Yang : Privacy risks of general-purpose language models. *In 2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331. IEEE, 2020.

[58] Dino Pedreshi, Salvatore Ruggieri et Franco Turini : Discrimination-aware data mining. *In Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 560–568, 2008.

[59] Jeffrey Pennington, Richard Socher et Christopher D Manning : Glove: Global vectors for word representation. *In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[60] Venkatadheeraj Pichapati, Ananda Theertha Suresh, Felix X. Yu, Sashank J. Reddi et Sanjiv Kumar : Adaclip: Adaptive clipping for private sgd. *ArXiv*, abs/1908.07643, 2019.

[61] Daniel Preoţiuc-Pietro, Vasileios Lampos et Nikolaos Aletras : An analysis of the user occupational class through twitter content. *In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1754–1764, 2015.

[62] Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky et Marc Najork : Privacy-adaptive BERT for natural language understanding. *arXiv preprint arXiv:2104.07504, accepted to CIKM 2021*, 2021.

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

[64] General Data Protection Regulation : Regulation eu 2016/679 of the european parliament and of the council of 27 april 2016. *Official Journal of the European Union*, 2016.

[65] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz et Michael Backes : Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *In Proceedings of the 26th Annual Network and Distributed System Security Symposium (NDSS)*, 2019.

[66] Gerard Salton et Christopher Buckley : Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988.

[67] Reza Shokri, Marco Stronati, Congzheng Song et Vitaly Shmatikov : Membership inference attacks against machine learning models. *In 2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2017.

[68] Congzheng Song et Ananth Raghunathan : Information leakage in embedding models. *In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390, 2020.

[69] Samuel Sousa et Roman Kern : How to keep text private? a systematic review of deep learning methods for privacy-preserving natural language processing. *Artificial Intelligence Review*, pages 1–66, 2022.

[70] Latanya Sweeney : k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[71] Latanya Sweeney : Only you, your doctor, and many others may know. *Technology Science*, 2015092903(9):29, 2015.

[72] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy et Samuel R Bowman : Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

[73] Yu-Xiang Wang, Borja Balle et Shiva Prasad Kasiviswanathan : Subsampled rényi differential privacy and analytical moments accountant. *In The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1226–1235. PMLR, 2019.

[74] T. Wolf, L. Debut, V. Sanh, J. Chaumond et A. Rush : Transformers: State-of-the-art natural language processing. *In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2020.

[75] Z. Xu, A. Aggarwal, O. Feyisetan et N. Teissier : A differentially private text perturbation method using a regularized mahalanobis metric. 2020.

[76] Qiao Xue, Youwen Zhu et Jian Wang : Mean estimation over numeric data with personalized local differential privacy. *Frontiers of Computer Science*, 16(3):1–10, 2022.

[77] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen et Han Yu : Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3):1–207, 2019.

[78] Samuel Yeom, Irene Giacomelli, Matt Fredrikson et Somesh Jha : Privacy risk in machine learning: Analyzing the connection to overfitting. *In 2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282. IEEE, 2018.

[79] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin et Tie-Yan Liu : Large scale private learning via low-rank reparametrization. *arXiv preprint arXiv:2106.09352*, 2021.

[80] Lei Yu, Ling Liu, Calton Pu, Mehmet Emre Gursoy et Stacey Truex : Differentially private model publishing for deep learning. *In 2019 IEEE Symposium on Security and Privacy (SP)*, pages 332–349. IEEE, 2019.

[81] Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun et Sherman SM Chow : Differential privacy for text analytics via natural text sanitization. *arXiv preprint arXiv:2106.01221, accepted to ACL-ICJNLP'21 Findings*, 2021.

[82] Benjamin Zi Hao Zhao, Mohamed Ali Kaafar et Nicolas Kourtellis : Not one but many tradeoffs: Privacy vs. utility in differentially private machine learning. *In Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop*, pages 15–26, 2020.

[83] Yuqing Zhu, Jinshuo Dong et Yu-Xiang Wang : Optimal accounting of differential privacy via characteristic function. *arXiv preprint arXiv:2106.08567*, 2021.

Appendix A

Mathematical Proof

A.1. DP Guarantee for SANTEXT and CusText

We give the mathematical proof for DP Guarantee in the followings. The main difference is their applicability in mathematical forms.

d_χ-privacy Guarantee for SANTEXT. The proof of SANTEXT [**81**] can provide $\epsilon \cdot d(x, x')$-DP protection relying on the triangle inequality of d within d_χ-privacy notion from original paper:

$$\frac{Pr[\mathcal{M}(x) = y]}{Pr[\mathcal{M}(x') = y]} = \frac{C_x \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x),\phi(y))}}{C_{x'} \cdot e^{-\frac{1}{2}\epsilon \cdot d_{\mathsf{euc}}(\phi(x'),\phi(y))}} \tag{A.1.1}$$

$$= \frac{C_x}{C_{x'}} \cdot e^{\frac{1}{2}\epsilon[d(x',y)-d(x,y)]} \tag{A.1.2}$$

$$\leq \frac{C_x}{C_{x'}} \cdot e^{\frac{1}{2}\epsilon d(x,x')} \tag{A.1.3}$$

$$= \frac{\sum_{y' \in \mathcal{Y}} e^{-\frac{1}{2}\epsilon d(x',y')}}{\sum_{y' \in \mathcal{Y}} e^{-\frac{1}{2}\epsilon d(x,y')}} \cdot e^{\frac{1}{2}\epsilon d(x,x')} \tag{A.1.4}$$

$$\leq e^{\frac{1}{2}\epsilon d(x,x')} \cdot e^{\frac{1}{2}\epsilon d(x,x')} \tag{A.1.5}$$

$$= e^{\epsilon d(x,x')} \tag{A.1.6}$$

where $C_x = (\sum_{y' \in \mathcal{Y}} e^{-\frac{1}{2}\epsilon d_{\mathsf{euc}}(\phi(x),\phi(y))})^{-1}$, and d_{euc} denotes Euclidean distance.

ϵ-DP Guarantee for CusText. Given the pre-defined sensitivity $\Delta u = 1$ and the constraint $\exists\ M \in \mathbb{R}$ s.t., $u(x,y) < M$, we show the proof of CusText satisfies ϵ-DP guarantee

with Exponential mechanism as below:

$$\frac{Pr[f_{\mathsf{sample}}(x)=y]}{Pr[f_{\mathsf{sample}}(x')=y]} = \frac{\frac{e^{\frac{\epsilon u(x,y)}{2\Delta u}}}{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x,y')}{2\Delta u}}}}{\frac{e^{\frac{\epsilon u(x',y)}{2\Delta u}}}{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x',y')}{2\Delta u}}}} \tag{A.1.7}$$

$$= e^{\frac{\epsilon\cdot(u(x,y)-u(x',y))}{2\Delta u}} \cdot \left(\frac{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x,y')}{2\Delta u}}}{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x',y')}{2\Delta u}}}\right) \tag{A.1.8}$$

$$\leq e^{\frac{\epsilon}{2}} \cdot e^{\frac{\epsilon}{2}} \cdot \left(\frac{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x,y')}{2\Delta u}}}{\sum_{y'\in\mathcal{Y}'} e^{\frac{\epsilon u(x,y')}{2\Delta u}}}\right) \tag{A.1.9}$$

$$= e^{\epsilon} \tag{A.1.10}$$

The proof, showing CusText ensures ϵ-DP, mainly relies on the triangle inequality of the score function $u(\cdot,\cdot)$.

To sum up, we can see the d_χ-privacy notion is only applicable for the similarity metrics satisfying triangle inequality of d, while the original ϵ-DP notion has no limitations. This is the motivation that we design to turn the CusText mechanism from satisfying d_χ-privacy to satisfying ϵ-DP.