Selective Differential Privacy for Language Modeling

Weiyan Shi[1], **Aiqi Cui**[1], **Evan Li**[1], **Ruoxi Jia**[2] **Zhou Yu**[1]
Columbia University[1],Virginia Tech[2]
{ws2634, ac4788, el3078}@columbia.edu, ruoxijia@vt.edu, , zy2461@columbia.edu

Abstract

With the increasing applications of language models, it has become crucial to protect these models from leaking private information. Previous work has attempted to tackle this challenge by training RNN-based language models with differential privacy guarantees. However, applying classical differential privacy to language models leads to poor model performance as the underlying privacy notion is over-pessimistic and provides *undifferentiated* protection for all tokens in the data. Given that the private information in natural language is sparse (for example, the bulk of an email might not carry personally identifiable information), we propose a new privacy notion, *selective differential privacy*, to provide rigorous privacy guarantees on the sensitive portion of the data to improve model utility. To realize such a new notion, we develop a corresponding privacy mechanism, Selective-DPSGD, for RNN-based language models. Besides language modeling, we also apply the method to a more concrete application – dialog systems. Experiments on both language modeling and dialog system building show that the proposed privacy-preserving mechanism achieves better utilities while remaining safe under various privacy attacks compared to the baselines. The data and code are released to facilitate future research[1].

1 Introduction

Language models have been widely used in various kinds of applications, such as Google Smart Compose, Amazon Alexa, and so on. However, these models are often trained on highly sensitive data such as emails and chat logs, while having the tendency to memorize the training data unintentionally (Carlini et al., 2019, 2020). Therefore, how to protect user privacy while preserving the model utility has become an increasingly important topic.

Several methods have been developed to protect the data, such as data anonymization, k-anonymity, and differential privacy (DP). Among them, DP has become a dominant privacy notion as it provides formal and provable guarantees for people to understand the privacy-utility trade-off (Figure 1). It works by carefully randomizing the algorithm so that the model does not rely too much on any single data point. However, traditional DP notion protects each data point as a whole regardless of the property of individual attributes inside the data (McMahan et al., 2018). Training large models with this overly pessimistic privacy notion could lead to poor model performance or even a non-convergent training process (Kerrigan et al., 2020). Our work is inspired by an important observation that in many scenarios including language modeling, private information is sparse, and not all attributes need to be protected. For example, for the sentence "My SSN is 123-45-6789", only the last token with the actual SSN number needs to be protected. But if we protect the entire sentence, we may fail to learn the underlying language pattern well.

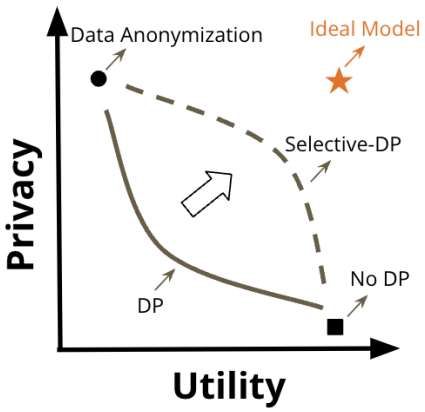

Figure 1: "Data anonymization" and training with "No DP" *cannot* provide knobs to adjust the trade-off between privacy and utility. Selective-DP improves the privacy-utility trade-off of traditional DP, to get closer to the ideal model with both high privacy and high utility.

To solve this problem, we propose a new DP notion, namely *selective differential privacy* (S-

[1] https://github.com/wyshi/lm_privacy

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 2848 - 2859

DP), to provide focused protection for sensitive attributes in a training record and improve model utility. We follow the traditional DP setup, where each training record is contributed by a different user. The key difference is that we consider only partial dimensions of a training record as sensitive, which better abstracts the language data privacy problems. Whether a given dimension in a record is sensitive is specified by a user-defined policy function that encodes application-specific privacy regulations, which gives users the freedom to protect any type of sensitive information according to the use cases. We also develop a corresponding privacy mechanism, Selective-DPSGD, for RNN-based language models under the new notion. Moreover, to process the variable-length sequences, we propose a batching method to group private and public tokens together and alternatively update them for a more efficient implementation of Selective-DPSGD.

One important concept to note is that, as shown in Figure 1, there is always a trade-off between privacy and utility: data anonymization achieves high privacy guarantee at the cost of low utility on private tokens; models trained without DP have high utility but low privacy. However, for both data anonymization and "no DP", there is no way to tune the trade-off, while DP-related methods provide knobs to adjust the privacy-utility trade-off. This paper proposes S-DP to improve the trade-off of canonical DP to get closer to the ideal model with both high-utility and high-privacy.

We evaluate S-DP on two tasks, 1) a language generation task and 2) a more concrete application of dialog systems. Besides reporting the model utility and theoretical privacy guarantees, we also empirically demonstrate their robustness under popular privacy attacks on language data (Carlini et al., 2019, 2020). The experiments suggest that training with Selective-DPSGD improves the model utility while remaining safe to the attacks.

Our contributions are as follows. First, we propose a **new** selective differential privacy **notion** that ensures targeted privacy protection for sensitive attributes, and a corresponding mechanism to realize the new S-DP notion for RNN-based models. Second, we propose a dialog dataset for future privacy research. Next, we show both theoretically and practically that our models are safe to attacks with improved utilities on both the language generation task and the dialog system application. Moreoever, we discuss the case of imperfect policy function, and compare S-DP with data anonymization to show that S-DP achieves better utilities when the policy function is imperfect. We also show preliminary results on contextual policy functions and Transformer models. In the era of information explosion and large-scale pretraining, protecting data privacy becomes more and more important. With S-DP, we march one step closer towards more privacy-preserving and high-performing language models and hope to inspire more research in this direction in the NLP community. Moreover, despite our focus on language-related applications in this paper, the proposed S-DP notion could be useful for a much broader range of applications where only partial data require privacy protection.

2 Related Work

Language modeling is a key research problem in NLP. However, although language models are often trained on sensitive data such as emails, most related studies focus on improving model without considering privacy, leaving the models vulnerable to attacks. For example, Carlini et al. (2019) showed that it is possible to infer a secret in the training data by attacking published language models. Therefore, it is of great importance to introduce privacy protection mechanisms to the NLP community and train the models in a much safer way.

Differential privacy (DP) (Dwork et al., 2014) has been applied to various domains (Cortés et al., 2016; Abowd, 2018). Abadi et al. (2016) introduced DPSGD to train DP-protected deep-learning models. PATE (Papernot et al., 2018) leveraged knowledge distillation to train differentially private models. But DP-protected algorithms suffer from low utility, so the DP notion often needs to be adjusted according to the applications: Ebadi et al. (2015) proposed a personalized DP notion to provide different levels of protection for different users; Doudalis et al. (2017) developed one-sided DP to protect sensitive users only. We also propose a new Selective-DP notion to protect only the sensitive attributes in a record to improve utility.

Recently, DP has also been applied to NLP tasks (Fernandes et al., 2019; Xu et al., 2020; Hathurusinghe et al., 2021; Sasada et al., 2021). For example, McMahan et al. (2018) proposed DP-FedAvg and DP-FedSGD to train RNN language models with user-level privacy guarantees. Adelani et al. (2020) developed a probabilistic text de-identification algorithm with formal privacy guar-

antees. Different from existing work that directly applied DPSGD and provided undifferentiated protection for all training examples, we propose a new privacy notion and a corresponding mechanism to protect the sensitive portion of the data in centralized learning. Such a new notion can be easily adapted to federated learning settings as well.

There is also a line of work that attempts to improve differentially private deep learning performance via modifying training process (Wang et al., 2021, 2019; Thakkar et al., 2019; Lee and Kifer, 2020) or model architecture (Papernot et al., 2020). Our work is complementary to this line of work as we propose a new privacy notion. Particularly, our work can be combined with the aforementioned methods to further improve model utility.

3 Backgrounds

We will introduce language modeling and differential privacy as preliminaries in this section.

Language Modeling. Consider a text sequence that consists of multiple tokens, i.e., $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$, where x_i is the i-th token. The goal of language modeling is to learn the probability of the sequence $p(\boldsymbol{x})$, which can be factorized with the chain rule as in Equation (1). Given a corpus $D = \{\boldsymbol{x}^1, \ldots, \boldsymbol{x}^{|D|}\}$, we train a neural network (e.g., RNN) parameterized by θ to learn $p(\boldsymbol{x})$ by minimizing the negative log-likelihood over D with the loss function in Equation (2).

$$p(\boldsymbol{x}) = \prod_{i=1}^{n} p(x_i|\boldsymbol{x}_{<i}), \tag{1}$$

$$\mathcal{L}(D) = -\sum_{t=1}^{|D|}\sum_{i=1}^{n_t} \log p_\theta(x_i^t|\boldsymbol{x}_{<i}^t) \tag{2}$$

Differential Privacy (DP) (Dwork et al., 2014). A differentially private algorithm ensures that its output cannot help much to distinguish whether an individual record is contained in the input dataset. In other words, DP hides the presence of individual records. The formal definition is as follows.

Definition 1. *(Differential Privacy). Given a domain $\mathcal{D}$, any two datasets $D, D' \subseteq \mathcal{D}$ that differ in exactly one record are called neighboring datasets. A randomized algorithm $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ is (ϵ, δ)-differential private if for all neighboring datasets D and D' and all $T \subseteq \mathcal{R}$,*

$$\Pr[\mathcal{M}(\mathcal{D}) \subseteq T] \leq e^\epsilon \Pr[\mathcal{M}(\mathcal{D}') \subseteq T] + \delta.$$

4 Selective Differential Privacy

Canonical DP notion treats all records as sensitive. Prior work has studied variants of DP notions, such as personalized DP (Jorgensen et al., 2015) and one-sided DP (Doudalis et al., 2017), to exploit different privacy levels between records. However, existing privacy notions do not allow different attributes in a given record to have different privacy levels, which could otherwise potentially enable additional utility gains, especially for NLP tasks where private attributes are sparse. Hence, we propose a new privacy notion–*selective differential privacy*–to distinguish between private and non-private attributes inside one data point with a *policy function* and protect sensitive part of one data point.

Definition 2. *(Policy Function). A policy function $F : \tau \to \{0, 1\}^{n_r}$ denotes which attributes of a record $r \in \tau$ are sensitive ($F(r)_i = 0$) or non-sensitive ($F(r)_i = 1$), where n_r is the number of attributes in r. Note that n_r depends on the record and is not a fixed number.*

In practice, users have the freedom to define the policy function to encode specific privacy regulation and protect any sensitive attributes according to the applications. The protected sensitive attribute types are unlimited, can be entities (e.g., name, emails, etc), contextual (e.g., health-related information, speaking style, etc), and so on. For example, users can design a conservative policy function that protects selected complete sentences if necessary. The form of the policy function is also unlimited, and could be neural networks, regex, and so on. Please refer to Section 6 for contextual policy functions.

In the case of language modeling, each record is a text sequence $\boldsymbol{x}$, each attribute is a token x_i in x and $F(\boldsymbol{x})$ is a bit vector indicating which tokens contain private information. We define neighboring datasets under our new privacy notion as follows.

Definition 3. *(F-Neighbors). D, D' are two datasets and F is a policy function. D' is a F-neighbor of D if and only if $\exists r \in D$ s.t., $F(r)$ contains at least one private attribute, $\exists r' \in D'$ s.t., $F(r)$ and $F(r')$ differ by at least one private attribute, and $D' = D \backslash \{r\} \cup \{r'\}$. We denote $D' \in N_F(D)$.*

Under this definition, the dataset containing "My ID is 123" and the dataset containing "My ID is 456" are neighbors; but the dataset with "Hello

there" and the dataset with "Hi there" are not neighbors since they do not contain private information.

Definition 4. *(Selective Differential Privacy). Given a policy function F, a randomized algorithm $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ satisfies (F, ϵ, δ)-selective differential privacy if for $\forall D, D' \in N_F(D)$, and $\forall T \subseteq \mathcal{R}$,*

$$\Pr[\mathcal{M}(D) \subseteq T] \leq e^{\epsilon} \Pr[\mathcal{M}(D') \subseteq T] + \delta.$$

Essentially, S-DP also provides an indistinguishability property similar to canonical DP, but only for the sensitive attributes in a record. S-DP does not constrain the information leakage of non-sensitive attributes as long as the privacy of the sensitive attributes is preserved. Thus, S-DP protects privacy for sensitive attributes in the worst case (i.e., the attacker may have knowledge about everything except the targeted sensitive attribute.)

4.1 Selective Privacy Mechanism

With the new S-DP notion, the next step is to develop a corresponding privacy mechanism to train models that realize the new notion. Privacy mechanisms usually work by adding noise to the models to protect the data, such as Laplace mechanism (Laplace noise) and Gaussian mechanism (Gaussian noise). Abadi et al. (2016) proposed DPSGD that adds Gaussian noise to the gradients and applies stochastic gradient descent (SGD) to train private deep learning models. In this work, we develop *Selective-DPSGD*, shown in Figure 2 and Algorithm 1, to train RNN-based language models that achieve S-DP. The basic idea is to first determine the private attributes with the policy function, then decide which model variables are related to the private attributes, and finally apply regular SGD on non-private variables, and DPSGD (Abadi et al., 2016) on the private variables. We choose RNNs because they are widely used in industry, e.g., Ramaswamy et al. (2020) discussed how to train private production language models with RNNs.

We need to first decide the variables related to the private tokens. RNN uses a hidden state $\boldsymbol{h}_i$ to encode the context, and outputs a distribution $\boldsymbol{p}_i$ over a vocabulary set V, as shown in Equation (3). If x_i is private, then $\boldsymbol{h}_i$, $\boldsymbol{p}_i$, and $\mathcal{L}_i$ are all private; besides, to calculate $\mathcal{L}_{i-1}$, we need to access the ground truth next token x_i, so $\mathcal{L}_{i-1}$ is also private. The private variables are all in red in Figure 2.

$$\boldsymbol{h}_i = RNN(\boldsymbol{h}_{i-1}, x_i) \quad (3)$$

$$\boldsymbol{p}_i = p_\theta(V|\boldsymbol{x}_{<i}) = Softmax(g(\boldsymbol{h}_i)) \quad (4)$$

$$\mathcal{L}_i = -\log p_\theta(x_{i+1}|\boldsymbol{x}_{<i+1}) \quad (5)$$

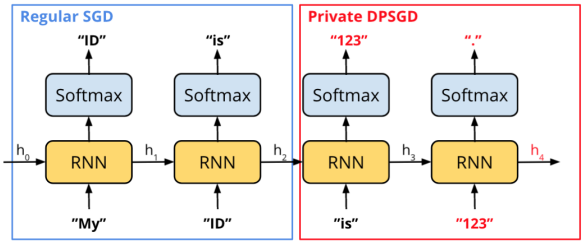

Figure 2: All private variables are in red. We apply regular SGD on non-private variables and DPSGD on private variables in Selective-DPSGD.

Algorithm 1 outlines the steps in *Selective-DPSGD*. Given a dataset D, we apply a policy function F to obtain a bit matrix $P = F(D)$ that indicates which tokens are private. At each step, we take a random batch B, and use P to split B into a sequence of non-private and private tuples $\{(B_{np,i}, B_{p,i})\}$; then we apply SGD (regular update) on $B_{np,i}$ and DPSGD (private update) on $B_{p,i}$ alternatively, to update and protect privacy. Note that besides noise in the gradients, we also clip and add noise to the hidden state $\boldsymbol{h}_i$ if it is private. The reason is that in RNN, if x_i is private, $\boldsymbol{h}_i$ also contains private information (as shown above), and is directly passed to the next regular update step and cannot be protected by the noise in the gradients. So it is important to add noise to protect the private information in $\boldsymbol{h}_i$. Since DPSGD adds noise to the gradients, $\mathcal{L}$ and $\boldsymbol{p}_i$ used to calculate the loss are protected by the noise in the gradients. In this way, all private variables are protected.

Privacy Guarantee. In Section A.3, we prove that the composition of the series of noise-adding operations ensures S-DP for Selective-DPSGD.

5 Experiments

We conduct our experiments on two datasets: 1) a traditional text corpus for language modeling, and 2) a dialog dataset for a more concrete application of dialog systems. Below are the dataset details.

WikiText-2. To minimize real-world harm, we choose the already-public WikiText-2 (Merity et al., 2017). It contains articles from Wikipedia with potentially sensitive information, and is a classical dataset for language modeling. For simplicity, we treat all the digits as privacy information. So the policy function F is a digit detector: if the token is a digit, F will output 0, otherwise, 1.

CUSTOMERSIM. With the emergence of virtual assistants, more and more private information is being exchanged during daily interactions. So we

Algorithm 1 Selective-DPSGD

Input: Dataset D with N examples, policy function F, privacy bit matrix $P = F(D)$, max sequence length K, loss function $\mathcal{L}(\theta)$.
Parameters: learning rate η, noise multiplier σ, gradient norm bound C, group size L.
for t=1,2,... **do**
  Take a random batch B of max sequence length K, with sampling probability L/N
  Using P, split B into a sequence of non-private and private tuples $\{(B_{np,i}, B_{p,i})\}$
  Initialize $\boldsymbol{h} = \vec{0}$
  for i=1,2,... **do**
    1) Regular update
    $\mathcal{L}, \boldsymbol{h}$ = Model($B_{np,i}$, $\boldsymbol{h}$)
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
    2) Private update
    $\mathcal{L}, \boldsymbol{h}$ = Model($B_{p,i}$, $\boldsymbol{h}$)
    Calculate sample gradient
    For each $x_j \in B_{p,i}$, compute $\boldsymbol{g}(x_j) \leftarrow \nabla_\theta \mathcal{L}(\theta, x_j)$
    Clip gradient
    $\boldsymbol{g}(x_j) \leftarrow \boldsymbol{g}(x_j) / \max(1, \frac{\|\boldsymbol{g}\|_2}{C})$
    Add Noise
    $\boldsymbol{g}(x_j) \leftarrow \frac{1}{|B_{p,i}|}(\sum_j \boldsymbol{g}(x_j) + \sigma C \cdot \mathcal{N}(0, \mathbf{I}))$
    Descent
    $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta)$
    Clip hidden states
    $\boldsymbol{h}(x_j) \leftarrow \boldsymbol{h}(x_j) / \max(1, \frac{\|\boldsymbol{h}\|_2}{C})$
    Add Noise
    $\boldsymbol{h}(x_j) \leftarrow \boldsymbol{h}(x_j) + \sigma C \cdot \mathcal{N}(0, \mathbf{I})$
  end for
end for

also apply S-DP to build dialog systems in the customer service domain. Using real dialog transcripts may lead to real-world harm, so we simulate a dialog dataset, CUSTOMERSIM, with synthetic user information. The dialogs are simulated with fixed agendas and template utterances (Zhao and Eskenazi, 2018). We treat user name, address, phone number, order, and tracking number as sensitive information, and use regex to build a policy function to detect them. Table 1 shows one example dialog.

Note that although we use digits and names as running examples for sensitive information, S-DP can protect any sensitive attributes specified by the policy function. Building better policy functions is orthogonal to S-DP, and thus beyond the scope of this paper. Any improvements on policy functions are compatible with S-DP to achieve better results.

Model training details. We use one-layer LSTMs with an embedding size of 200 and a hidden size of 200, and a BPE tokenizer (Sennrich et al., 2016) to avoid information leakage from the tokenizer: with BPE, a secret "1234" will be split into multiple tokens, e.g., "12" and "34", while traditional tokenizer will release "1234" in the dictionary. All private states (hidden, cell states) in the LSTMs are protected.

CUSTOMERSIM

Role	Utterance
SYS	Hello, I am the customer support bot. What can I do for you?
USR	Hello robot. Could you please help me track my package?
SYS	Please provide your full name.
USR	Sure, Betty Sims.
SYS	Could you please confirm your shipping address?
USR	Yea sure, 2241 Fitzgerald Viaduct Brownview, OK 28304.
SYS	Track your order using your tracking number, FH6F6GMMF4. Are you happy about my answer?
USR	That's it.

Table 1: An example dialog in CUSTOMERSIM.

Baselines. We have two baselines, one without DP ("No-DP"), and the other trained with DPSGD ("DPSGD"). We refer to our models trained with S-DPSGD as "S-DPSGD". "No-DP" is simply an LSTM optimized with a regular SGD and a starting learning rate (lr) of 20. The learning rate was annealed and decreased as training proceeded. "DPSGD" is optimized with DPSGD and a starting learning rate of 0.05. All the models are trained five times to reduce randomness, and the parameters are tuned on the validation set. We compare with DPSGD because it's the backbone of most of existing DP learning algorithms. Existing modifications of DPSGD are mainly focused on optimization algorithms and objectives, thus are compatible with our work that tailors the privacy notions to realistic privacy needs in the NLP context.

5.1 Evaluation

We evaluate both the language models' utilities and privacy protection levels. We use perplexity (PPL) and the top-1 next word prediction accuracy (AccT1) to measure model utility. To measure privacy protection levels, besides reporting the theoretical privacy budget ϵ and δ, we also perform various practical attacks on the trained models and report how successful the attacks are against different techniques. We compare the performance of our proposed privacy-preserving learning technique and the baselines in terms of the privacy-utility trade-off. Specifically, we compare the utility between different techniques at a *given* privacy protection level, or vice versa.

	WikiText-2						CUSTOMERSIM					
Model	**PPL**	**AccT1**	σ	C	ϵ	δ	**PPL**	**AccT1**	σ	C	ϵ	δ
LSTM, No-DP	60.98 ± 0.48	0.36 ± 0.00	-	-	-	-	3.06 ± 0.01	0.75 ± 0.00	-	-	-	-
DPSGD	305.86 ± 3.00	0.27 ± 0.00	0.50	0.10	4.89	8e-5	11.82 ± 0.76	0.70 ± 0.01	0.60	0.01	2.51	8e-5
S-DPSGD	160.04 ± 4.86	0.31 ± 0.00	0.50	1e-3	4.91	8e-5	10.42 ± 0.91	0.69 ± 0.01	0.70	5e-3	2.74	8e-5

Table 2: Model utility and privacy guarantee on WikiText-2 (left) and CUSTOMERSIM (right). **PPL**: Perplexity on the test set. **AccT1**: Top-1 next word prediction accuracy. σ: Noise multiplier in the Gaussian noise. C: Clipping threshold. ϵ, δ: Privacy guarantee in (ϵ, δ)-privacy.

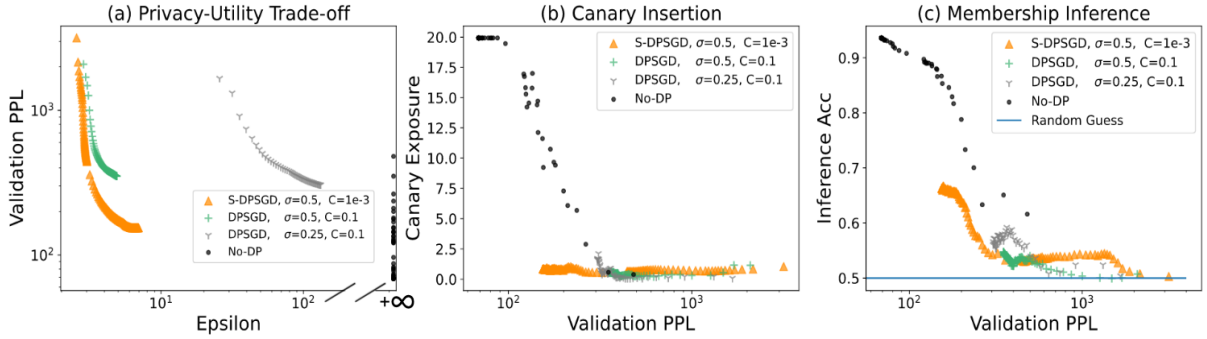

Figure 3: Privacy-utility trade-off, canary insertion attack and membership inference attack on WikiText-2.

5.1.1 Attack Details

We perform two types of attacks: 1) canary insertion and 2) membership inference.

Canary insertion is proposed by Carlini et al. (2019). It first inserts random sequences called canaries into the training dataset, then trains the model, and finally calculates the following *exposure* for the inserted canaries to measure if the model will unintentionally memorize these canaries. Canaries are of a specific format, e.g., s="The number is ◌◌◌◌◌◌", where ◌ are filled with random values from a randomness space $\mathcal{R}$ such as a space of digits from 0 to 9. To obtain the *exposure*, we enumerate all possible sequences in the specified form, and calculate the negative log-rank with an additional constant, as shown below. Lower exposure indicates the model is more private. In our setting, we insert the secret "My ID is 341752" into the training data for 10 times to make the differences between models more salient.

Definition 5. *Given a canary $s[r]$, a model with parameters θ, and the randomness space $\mathcal{R}$, the **exposure** of $s[r]$ is*

$$\textbf{\textit{exposure}}_\theta = \log_2 |\mathcal{R}| - \log_2 \textbf{\textit{rank}}_\theta(s[r])$$

Membership Inference is a widely used attack method that identifies if a given sample is a member of the training dataset. Lower inference accuracy means that it is harder to infer a member from the model and thus the model is safer. Carlini et al. (2020) proposed an advance membership inference attack for language models. The basic idea is to calculate the given samples' perplexities under the model, rank them and choose the ones with the lowest perplexities (highest likelihood by the model). In our experiments, we randomly select 500 protected secrets from the training set, and randomly generate 500 samples of similar format, to form a dataset for the membership inference attack, so a random guess would give us an accuracy of 50%. For WikiText-2, the secrets are digit sequences; for CUSTOMERSIM, customer names are the secrets.

5.2 WikiText-2 Results

Model utility and privacy guarantee. The left part of Table 2 shows different models' utilities and privacy guarantees on WikiText-2 and Figure 3(a) shows the privacy-utility trade-off, where utility is represented by the validation perplexity (lower PPL=higher utility) and the privacy budget is represented by the epsilon (smaller ϵ=more privacy). Although the definitions of ϵ in DPSGD and S-DPSGD are different, the ϵ in both cases provides a *tight* theoretical upper bound on how much the sensitive attributes are leaked through the trained models. Hence, the ϵ associated with two algorithms are comparable. Given a fixed privacy budget ϵ, we want to achieve a higher model utility. Because

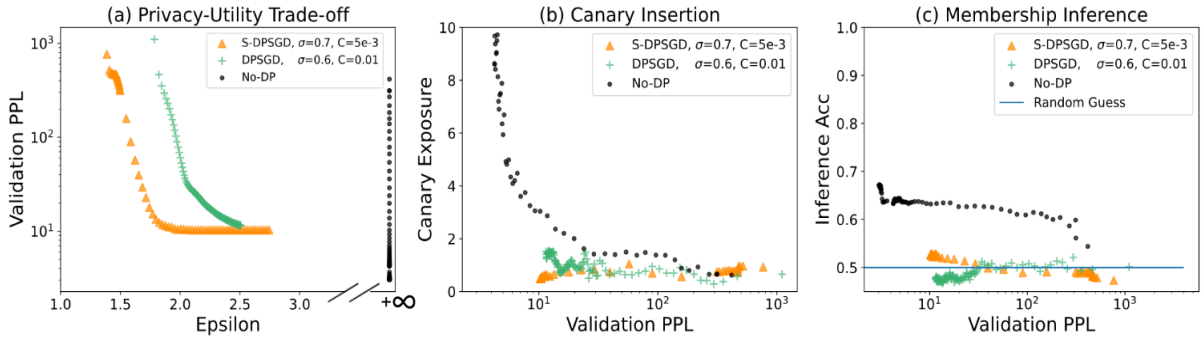

Figure 4: Privacy-utility trade-off, canary insertion attack and membership inference attack on CUSTOMERSIM.

the privacy budget $\epsilon = +\infty$ for the No-DP model, "No-DP" is represented by the vertical black line on the far right in Figure 3(a), and it achieves the best average PPL of 60.98 on the test set. The orange line is our S-DPSGD model, and it achieves the second-best average test PPL of 160.04 with $\epsilon = 4.91$. With a similar $\epsilon = 4.89$, DPSGD has the worst average test PPL of 305.86, much worse than S-DP because canonical DP notion protects the whole data and is over-pessimistic (see Section A.4 for models with different parameters). The gray line is for DPSGD with a smaller $\sigma = 0.25$, a convergent PPL of 266.6 and a final $\epsilon = 132.73$. Compared to DPSGD with $\sigma = 0.25$, it achieves a better PPL but with a much higher cost of privacy leakage (larger ϵ). But with S-DPSGD, we can also achieve lower PPL without hurting privacy.

Attack results. Figure 3(b) and 3(c) show the canary insertion attack and membership inference attack results on WikiText-2. The x-axis is the models' utilities measured by validation PPL, and the y-axis is the exposure and membership inference accuracy indicating the success level of the attacks. Lower exposure and lower accuracy indicate a safer model. We want to see at a given robustness level to the attacks, which models can achieve lower perplexity, i.e., higher utility.

For canary insertion attack, although "No-DP" achieves lower perplexity, its exposure can go up to 20, indicating that the inserted canary could be easily revealed by the attackers. If we compare No-DP with S-DPSGD with similar utilities, S-DPSGD is always below No-DP, meaning S-DPSGD achieves much smaller exposure, and hence a safer model with similar utility. Comparing DPSGD and S-DPSGD, we find that S-DPSGD achieves much better model utility at a given exposure.

For membership inference attack, we draw a horizontal line of 0.5 to show the random guess performance. Again, S-DPSGD is always below No-DP, showing that with similar utilities, models trained with S-DPSGD are safer than No-DP under the membership inference attack. As mentioned earlier, DPSGD with $\sigma = 0.5$ (green) and S-DPSGD (orange) have similar privacy budget (ϵ=4.89 and 4.91 respectively). Comparing these two, we see that given a similar privacy budget, S-DPSGD converges to a much lower perplexity while remaining safe to the attack and thus achieves a wider range for the privacy-utility trade-off tuning. We also observe that for "No-DP", the inference accuracy can go up to 90%, suggesting that language models without DP protection are vulnerable to attacks.

5.3 CUSTOMERSIM Results

This section shows the results on CUSTOMERSIM.

Model utility and privacy guarantee. The right part of Table 2 and Figure 4(a) show the privacy-utility trade-off for CUSTOMERSIM. Because the dialogs are simulated with templates and relatively simple, the perplexity can be as low as 3.06 for No-DP. S-DPSGD still achieves better perplexity than DPSGD (10.42 vs. 11.82) with similar ϵ, but the gap is smaller compared to WikiText-2 because there are more sensitive tokens in CUSTOMERSIM (18.3%) than WikiText-2 (2.8%), and the advantage of protecting selective tokens only is not as big.

Attack results. Figure 4(b) and 4(c) show the results of the canary insertion and membership inference attack on CUSTOMERSIM respectively.

For canary insertion, we observe that given the same utility, S-DPSGD achieves lower exposure than No-DP and DPSGD. Although the improvement seems small on absolute values, we should note that exposure is on log-scale, so the improvement on relative rank is also on log-scale (e.g., for exposure=1.2, the rank of the canary is 429881; for exposure=0.2, the rank is 858215, roughly twice).

The membership inference accuracy is only a little better than random guess probability (60+%), so it is not successful on CUSTOMERSIM. One potential reason could be that customer names only appear once in each dialog and can be very similar to each other (e.g., Emily, Emilya, and Emilyah). We leave it as future work to develop better membership inference attacks towards similar secrets. Under the failed attack, S-DPSGD is still better than No-DP. It is not feasible to compare S-DPSGD and DPSGD as both are close to random guess.

5.4 Data Anonymization and Selective-DP

Data anonymization (or data de-identification) has been widely used to protect data privacy, where the sensitive tokens are masked with special tokens such as "<num>". Both data anonymization and S-DP target protection towards sensitive attributes, and rely on a policy function to detect the private information. However, they are different in that data anonymization masks the sensitive attributes completely so nothing can be learned from them, while S-DP noises the private portion and provides a tunable way to adjust the privacy-utility trade-off, evidenced by experiments in Section 5.4.1.

One common problem for both methods is that the policy function is not guaranteed to be perfect and can miss to detect some sensitive information. So we also compare their performance when the policy function is imperfect in Section 5.4.2.

5.4.1 S-DP achieves better utility

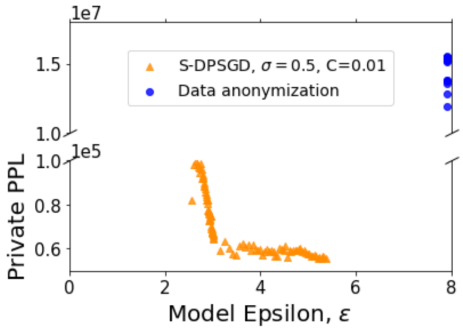

Figure 5: Perplexity on private tokens over ϵ for data anonymization and S-DPSGD.

We mask all the detected digits by "<num>" to train a data anonymization baseline on WikiText-2, calculate the perplexity on the private tokens, and present the result in Figure 5. The x-axis is the privacy budget ϵ, and the y-axis is the private-token perplexity. For data anonymization, the dots are all on the far right because $\epsilon{=}+\infty$. The first observation is that since data anonymization simply masks the sensitive tokens, it fails to learn anything about them, resulting in a much worse PPL on private tokens. This makes S-DP a good alternative to data anonymization because S-DP can still learn certain patterns from the noised sensitive tokens. Also, Figure 5 shows that for S-DP, there is a trade-off between ϵ (privacy) and private-token PPL (utility), so we could tune ϵ to achieve a better private-token PPL, while for data anonymization, there is no way to tune the parameters for better model utilities. More concretely, with proper ϵ and δ, S-DP might learn the structure of sensitive attributes (e.g., XXX-XX-XXXX for SSN) without knowing the exact value, or even learn the distribution of values for each digit, and such knowledge could be useful for data analytics. But for data anonymization, the model either sees the digit or doesn't see it, so there is no knob to tune the privacy-utility trade-off.

5.4.2 Imperfect Policy Function

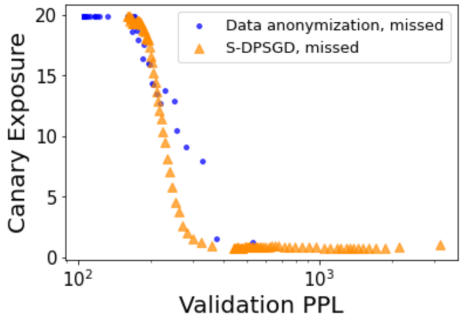

Figure 6: Canary insertion attack for data anonymization and S-DPSGD when missing the canary.

Now we discuss the performance of both methods when the policy function is imperfect. We still use WikiText-2 with the secret "My ID is 341752". The policy function fails to detect "341752" as a secret. For data anonymization, all the detected digits are masked by "<num>"; for S-DP, we apply S-DPSGD to noise the detected digits.

Figure 6 shows the exposure of the missed secret 341752. When the perplexity gets lower, the model becomes better at remembering the details of the training data, so the exposure of both models becomes high. But when the perplexity is around the middle area, S-DP has lower exposure than data anonymization, meaning it's safer to attacks.

Note that the risk with imperfect policy functions is common to many privacy-preserving techniques, and how to build better policy functions is orthog-

onal to this work on S-DP, and thus beyond the scope of this paper. Improvements on policy functions (better sensitive information detection) are compatible with S-DP and can be used to further improve the results. We are also actively working on this topic in parallel to the S-DP work.

6 Preliminary Results on Contextual Policy Function and Transformer

Model	Policy function	Portion	ϵ	PPL
GPT2 + no DP	-	-	-	20.47
DPSGD	-	-	2.58	27.05
Redacted	Noncontextual:	16.40%	-	24.30
S-DP	All entities		2.58	22.56
Redacted	Contextual: entities, sub,	34.80%	-	38.66
S-DP	obj, propon, pron		2.48	25.61

Table 3: Preliminary results on different policy functions and Transformers.

In this section, we present the preliminary results on different policy functions and large Transformer models (Vaswani et al., 2017) on WikiText-2.

We design two policy functions: one is non-contextual and protects all the 18 named entities detected by spacy (Honnibal and Montani, 2017) such as person, date, locations, etc (16.4% tokens)[2]; the other one is contextual that protects all the entities plus subjects, objects, proper noun and pronouns of all the sentences (34.8% tokens). We use these two policy functions to redact the WikiText-2 D and obtain a redacted version D'. We first fine-tune a GPT2-small model (Radford et al., 2019) on D' (denoted as "redacted model"), and then further fine-tune this redacted model on the original D (Shi et al., 2022). The results are in Table 3. The redacted models trained on the redacted data D' achieve 24.30 and 38.66 in perplexity for the two policy functions respectively. If we fine-tune these two models on the original private data D with DPSGD, we can further improve the perplexity to 22.56 and 25.61, while the state-of-the-art DP language models only achieve 27.05 with similar privacy budget. These results show that our S-DP notion is promising in boosting utility of privacy-preserving language models even if one-third of the tokens are considered sensitive.

[2]The full list of entities is available here https://spacy.io/usage/linguistic-features#named-entities

7 Conclusions

To conclude, we develop a new privacy notion, *selective differential privacy* (S-DP), to improve model utility while providing rigorous privacy guarantees for the sensitive portion of the data. We also develop a privacy mechanism, Selective-DPSGD, to achieve the new S-DP notion for RNNs. We experiment with WikiText-2 and a synthetic customer service dialog dataset. Results on both tasks show that models trained with S-DP achieve better utilities than traditional DP, and are more robust under various attacks than models without DP protection. With S-DP, we march one step closer towards safer and better language models and hope to inspire more related research. Moreover, S-DP could be applied to domains beyond NLP where only partial data require protection, such as image recognition. Please see Section A.1 for ethical consideration.

References

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

John M Abowd. 2018. The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2867–2867.

David Ifeoluwa Adelani, Ali Davody, Thomas Kleinbauer, and Dietrich Klakow. 2020. Privacy guarantees for de-identifying text transformations. *INTERSPEECH*.

Borja Balle, Gilles Barthe, and Marco Gaboardi. 2018. Privacy amplification by subsampling: Tight analyses via couplings and divergences. *Advances in Neural Information Processing Systems*, 31.

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

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2020. Extracting training data from large language models. *arXiv preprint arXiv:2012.07805*.

Jorge Cortés, Geir E Dullerud, Shuo Han, Jerome Le Ny, Sayan Mitra, and George J Pappas. 2016. Differential privacy in control and network systems. In

2016 IEEE 55th Conference on Decision and Control (CDC), pages 4252–4272. IEEE.

Stelios Doudalis, Ios Kotsogiannis, Samuel Haney, Ashwin Machanavajjhala, and Sharad Mehrotra. 2017. One-sided differential privacy. *Proceedings of the VLDB Endowment*.

Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407.

Hamid Ebadi, David Sands, and Gerardo Schneider. 2015. Differential privacy: Now it's getting personal. *Acm Sigplan Notices*, 50(1):69–81.

Natasha Fernandes, Mark Dras, and Annabelle McIver. 2019. Generalised differential privacy for text document processing. In *International Conference on Principles of Security and Trust*, pages 123–148. Springer, Cham.

Rajitha Hathurusinghe, Isar Nejadgholi, and Miodrag Bolic. 2021. A privacy-preserving approach to extraction of personal information through automatic annotation and federated learning. *arXiv preprint arXiv:2105.09198*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Zach Jorgensen, Ting Yu, and Graham Cormode. 2015. Conservative or liberal? personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*, pages 1023–1034. IEEE.

Gavin Kerrigan, Dylan Slack, and Jens Tuyls. 2020. Differentially private language models benefit from public pre-training. In *Proceedings of the Second Workshop on Privacy in NLP*, pages 39–45.

Jaewoo Lee and Daniel Kifer. 2020. Differentially private deep learning with direct feedback alignment. *arXiv preprint arXiv:2010.03701*.

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

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. *ICLR*.

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

Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. 2018. Scalable private learning with pate. *ICLR*.

Nicolas Papernot, Abhradeep Thakurta, Shuang Song, Steve Chien, and Úlfar Erlingsson. 2020. Tempered sigmoid activations for deep learning with differential privacy. *CoRR*, abs/2007.14191.

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

Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H Brendan McMahan, and Françoise Beaufays. 2020. Training production language models without memorizing user data. *arXiv preprint arXiv:2009.10031*.

Taisho Sasada, Masataka Kawai, Yuzo Taenaka, Doudou Fall, and Youki Kadobayashi. 2021. Differentially-private text generation via text preprocessing to reduce utility loss. In *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 042–047. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *ACL*.

Weiyan Shi, Si Chen, Chiyuan Zhang, Ruoxi Jia, and Zhou Yu. 2022. Just fine-tune twice: Selective differential privacy for large language models. *arXiv preprint arXiv:2204.07667*.

Om Thakkar, Galen Andrew, and H Brendan McMahan. 2019. Differentially private learning with adaptive clipping. *arXiv preprint arXiv:1905.03871*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*.

Bao Wang, Quanquan Gu, March Boedihardjo, Farzin Barekat, and Stanley J. Osher. 2019. DP-LSSGD: A stochastic optimization method to lift the utility in privacy-preserving ERM. *CoRR*, abs/1906.12056.

Wenxiao Wang, Tianhao Wang, Lun Wang, Nanqing Luo, Pan Zhou, Dawn Song, and Ruoxi Jia. 2021. Dplis: Boosting utility of differentially private deep learning via randomized smoothing. *arXiv preprint arXiv:2103.01496*.

Zekun Xu, Abhinav Aggarwal, Oluwaseyi Feyisetan, and Nathanael Teissier. 2020. A differentially private text perturbation method using regularized mahalanobis metric. In *Proceedings of the Second Workshop on Privacy in NLP*, pages 7–17.

Tiancheng Zhao and Maxine Eskenazi. 2018. Zero-shot dialog generation with cross-domain latent actions. *SIGDIAL*.

A Appendix

A.1 Ethical Consideration

Data Usage. To minimize real-world harm, we choose WikiText-2 since it is already public and widely used, and synthesize the dialogs as well as the personal information in the CUSTOMERSIM datasets in our experiments. For future research, we plan to continue using public or synthetic datasets to prevent real-world data leakage.

Application. Our work addresses the problem of data privacy protection and can be applied in different applications. The attacks used in our study are well-known standard attacks tailored for our specific tasks, so it's hard to generalize and misuse them to attack other language models. We will release the code so that people can have access to the various algorithms and protect their own data.

A.2 Limitations

There are many spaces for improvements for this S-DP work. For instance, when the policy function fails to detect sensitive attributes, their privacy may not be guaranteed, and therefore, we plan to develop better policy functions and employ privacy amplification (Balle et al., 2018). Also, besides explicit private information like names, we plan to protect sensitive context such as "I have two kids", as these can often happen causally in dialogs but still reveal personal status. In Section 6 we show some preliminary results on protecting contexts and plan to further refine the contextual policy functions.

A.3 Privacy Analysis

We analyze the private guarantees of Selective-DPSGD in this section.

For any given dataset D, let $D_{i,j}$ denote the jth attribute of the i-th record. We abstract the gradient update and hidden state into a query function $f(x, w)$ which takes training data x and auxiliary information w as input. We introduce w as an additional input to f to model the dependence of the gradient update and hidden state on the model parameters at the previous rounds. We define the following two types of queries on the dataset.

- Type-1 query: the input x to f consists of only private attributes with respect to the policy function F
- Type-2 query: the input x to f consists of only non-private attributes with respect to the policy function F

Since S-DP only hides the presence of private attributes, type-2 query does not incur privacy loss.

The following theorem shows that if a type-1 query has the property that its output is bounded, then for arbitrary auxiliary input, adding Gaussian noise into the query can provide DP. The reason why we consider such queries is that clipped gradient and hidden state can be modeled as such queries. The reason for which we want to analyze DP guarantees under arbitrary auxiliary inputs is that at any given round, the model parameters resulting from previous rounds could be arbitrarily different. This is because the non-sensitive part of two F-neighboring datasets could be arbitrarily different.

Theorem 1. *Assume that* $\max_{x,w} \|g(x, w)\| \leq C$. *Then, for any arbitrary* w, *adding Gaussian noise* $\Delta = \mathcal{N}(0, \sigma^2)$ *proportional to* C *into* g *can ensure* (ϵ, δ)*-DP where* ϵ, δ *depends on* C *and* σ. *More formally, for all neighboring datasets* x *and* x' *and all* w, w',

$$\frac{P[g(x, w) + \Delta = r]}{P[g(x', w') + \Delta = r]} \leq e^{\epsilon} \quad \textit{w.p. } 1 - \delta \qquad (6)$$

The proof follows directly from the classic proof for DP guarantees of the Gaussian mechanism (Mironov, 2017; Dwork et al., 2014) by noticing the sensitivity of f is bounded by C.

The regular updates in Algorithm 1 take as input non-private data $B_{np,i}$. Hence, they are type-2 queries and do not incur extra privacy loss. The private updates in Algorithm 1 (i.e., gradient and hidden states) depend on private attributes and model parameters from previous rounds, and thus belong to the type-1 query. Moreover, they satisfy the bounded norm assumption in Theorem 1. We call the resulting query of adding Gaussian noise into a type-1 query with bounded norm property a *noisy type-1 query*. Overall, Algorithm 1 is essentially the composition of multiple type-1 and noisy type-2 queries. In the following, we will present a general result for the privacy guarantees resulting from the composition.

Theorem 2. *Let* f *be the composition of* k *queries:* $f_1, \ldots, f_k$, *which are either noisy type-1 or type-2 queries. Given a policy function* F, *let* $\mathbf{f}_p$ *denote the set of noisy type-1 queries. Let* $\mathbf{f}_{np}$ *denote the set of type-2 queries. Then, if* $\mathbf{f}_p$ *is* (ϵ, δ)*-DP,* f *is* (F, ϵ, δ)*-S-DP.*

Proof. Consider two selective F-neighboring datasets x and x'. Let x_i and x'_i be the subset of data utilized by f_i. x_i contains only private attributes when f_i is type-1 and contains only non-private attributes when f_i is noisy type-2. By the neighboring relation between x and x', x_i and x'_i are also selective F-neighbors when f_i is a type-1 query. In addition to the dataset, f_i takes the output of all the previous queries. For a fixed outcome $(y_1, \ldots, y_k)$ of f, we have

$$\frac{P[f_1(x_1, w_1) = y_1, \ldots, f_k(x_k, w_k) = y_k]}{P[f_1(x'_1, w'_1) = y_1, \ldots, f_k(x'_k, w'_k) = y_k]} \quad (7)$$

$$= \prod_{f_i \in \mathbf{f}_p} \frac{f_i(x_i, w_i)}{f_i(x'_i, w'_i)} \quad (8)$$

$$\leq e^{\epsilon} \quad \text{w.p. } 1 - \delta \quad (9)$$

as desired.

The equality in the second line is due to the fact that $\mathbf{f}_{np}$ does not incur privacy loss and the independence of randomness of each query given the output of all the previous queries. The inequality in the third line is due to the assumption that $\mathbf{f}_p$ is (ϵ, δ)-DP with arbitrary auxiliary input. □

Instantiating the type-1 and type-2 queries in Theorem 2 with the regular and private updates defined in Algorithm 1 yields the privacy guarantees for Algorithm 1. Theorem 2 provides a convenient way of calculating the S-DP guarantees for Algorithm 1: one can apply off-the-shelf privacy composition tools to calculate the overall DP guarantees for all the private updates and then the entire algorithm satisfies S-DP with the same values of ϵ and δ. Specifically, in this paper, we leverage moment accountant (Abadi et al., 2016) to calculate the DP guarantees for the composition of all privacy queries.

A.4 Models with Different Parameters

We plot the performances of models with different parameters in Figure 7. We find that fixing the noise multiplier $\sigma = 0.5$, the clipping threshold C has a big impact on the performance, and it cannot be too big (0.01) or too small (5e-6); if we fix C and change σ from 0.5 to 0.1, the perplexity will be lower but at a huge cost of the privacy budget σ. As expected, there is always a trade-off between the utility and the privacy spent, so we choose a balancing point with a reasonable utility and privacy guarantee ($\sigma = 0.5$ and C =1e-3) for the main experiments.

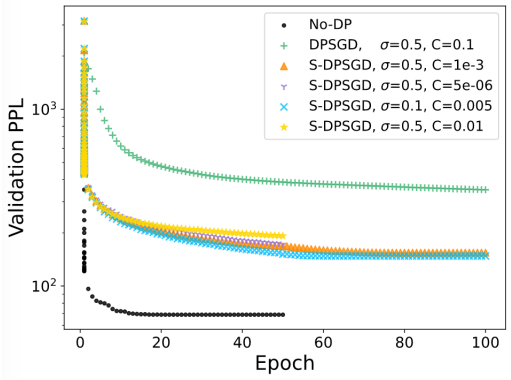

Figure 7: Validation perplexity over epochs on WikiText-2 for models with different parameters.

A.5 Membership Inference on CUSTOMERSIM

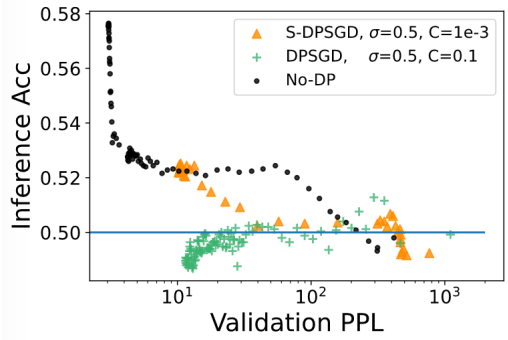

Figure 8: Original membership inference results on CUSTOMERSIM. The best inference accuracy is around 58%.

The original membership inference attack doesn't achieve good results on CUSTOMERSIM. Figure 8 shows the original membership inference result on CUSTOMERSIM. The best inference accuracy is around 58%. So we employ a more advanced version, where we first perform the attack on 1000 names, and then pick the best-predicted and worst-predicted names to form a new subset of 300 names to perform the attack again. But even for the advanced version, the inference accuracy is only a little better than random guess probability (60+%), so this attack is not successful on CUSTOMERSIM.