DP-BART for Privatized Text Rewriting under Local Differential Privacy

Timour Igamberdiev and **Ivan Habernal**
Trustworthy Human Language Technologies
Department of Computer Science
Technical University of Darmstadt
{timour.igamberdiev, ivan.habernal}@tu-darmstadt.de
www.trusthlt.org

Abstract

Privatized text rewriting with local differential privacy (LDP) is a recent approach that enables sharing of sensitive textual documents while formally guaranteeing privacy protection to individuals. However, existing systems face several issues, such as formal mathematical flaws, unrealistic privacy guarantees, privatization of only individual words, as well as a lack of transparency and reproducibility. In this paper, we propose a new system 'DP-BART' that largely outperforms existing LDP systems. Our approach uses a novel clipping method, iterative pruning, and further training of internal representations which drastically reduces the amount of noise required for DP guarantees. We run experiments on five textual datasets of varying sizes, rewriting them at different privacy guarantees and evaluating the rewritten texts on downstream text classification tasks. Finally, we thoroughly discuss the privatized text rewriting approach and its limitations, including the problem of the strict text adjacency constraint in the LDP paradigm that leads to the high noise requirement.[1]

1 Introduction

Protection of privacy is increasingly gaining attention in today's world, both among the general public and within the fields of machine learning and NLP. One very common methodology for applying privacy to an algorithm is Differential Privacy (DP) (Dwork and Roth, 2013). In simple terms, DP provides a formal guarantee that any individual's contribution to a query applied on a dataset is bounded. In other words, no individual can influence this query 'too much'.

One particular method of applying DP to the domain of NLP is *differentially private text rewriting*, in which an entire document is rewritten with DP guarantees by perturbing the original text representations. For instance, given a document "I would like to fly from Denver to Los Angeles this Thursday", the system may rewrite it as "Show me flights to cities in California this week". If one is training a model on intent classification for airline travel inquiry systems, either document would be a useful data point. In this way, we avoid using the original text that has uniquely identifiable qualities of a specific author, and instead create a privatized 'synthetic' example. This is in fact a form of local differential privacy (LDP), which is a stronger form of DP that is not limited to a specific dataset.

The benefits of an LDP text rewriting system are immense, where the output privatized dataset can be used for any downstream analysis. We also avoid the problem of having to manually determine what specific tokens in a document are private, applying LDP to the entire document. However, there is a significant difficulty in creating such a system, with a lot of perturbation required to achieve any reasonable privacy guarantees, leading to poor downstream utility. In addition, there are several issues in existing DP text rewriting systems, such as formal flaws having been discovered in their methodology (Habernal, 2021), older types of models used (e.g. single-layer LSTM, as in Krishna et al. (2021)), high privacy budgets, as well as a lack of transparency in the claimed privacy guarantees, outlined in Igamberdiev et al. (2022).

To address these issues, we propose **DP-BART**, a DP text rewriting system under the local DP paradigm that improves upon existing baselines and consists of several techniques that can be directly applied to a pre-trained BART model (Lewis et al., 2020), without having to design and train such a model from scratch. Despite being a large transformer architecture, it can be easily used for data privatization, not requiring many resources. Our methodology consists of a novel clipping method for the BART model's internal encoder representa-

[1] Our code is available at https://github.com/trusthlt/dp-bart-private-rewriting.

Findings of the Association for Computational Linguistics: ACL 2023, pages 13914–13934

tions, as well as a pruning and additional training mechanism that reduces the amount of DP noise that needs to be added to the data during the privatization process.

We summarize our contributions as follows. First, we present our **DP-BART** model and its related methodologies, aimed at reducing DP noise and reaching a better privacy/utility trade-off. For comparison, we use a reimplementation of the current primary baseline for this task, the ADePT model. Second, we run experiments to investigate the privacy/utility trade-off of these models, using five unique datasets that gradually increase in size, evaluating rewritten texts on downstream text classification tasks. Finally, we thoroughly examine the feasibility of the LDP text rewriting setting, investigating issues of the high noise requirement due to the strict text adjacency constraint, trade-offs between privacy and dataset size, what exactly is the object of privatization, required computational resources, as well as limitations of the approach as a whole and possible alternatives.

2 Related Work

We present a theoretical background on differential privacy, the BART model, and pruning for neural networks in Appendix A.

Applying differential privacy to neural network training and model publishing has converged to using a mainstream method, namely DP-SGD (Abadi et al., 2016). However, the task of text privatization is still broadly unexplored, with many unanswered questions remaining, such as dealing with the unstructured nature of text and explainability of the privacy guarantees provided to textual data (Klymenko et al., 2022). Mattern et al. (2022a) explored text rewriting with global differential privacy, sampling from a generative language model trained with DP.

There are only a few approaches that directly tackle the problem of differentially private text rewriting with LDP. Krishna et al. (2021) developed the ADePT system, which is an RNN-based text autoencoder that incorporates DP noise to its encoder output hidden state. As described by Habernal (2021), ADePT had a formal error in calculating the Laplace noise scale, which resulted in it violating differential privacy.

A more recent text rewriting system is DP-VAE (Weggenmann et al., 2022), which added constraints to the vanilla VAE model latent space (Kingma and Welling, 2014) to obtain a bounded sensitivity on its mean and variance parameters. Despite the high difficulties of the task, the paper reports surprisingly high performance for high privacy standards. Since their experimental description lacks some key details and the code base is not public, we cannot reproduce their approach.

In addition, there are a number of word-level DP systems (Feyisetan et al., 2019; Xu et al., 2020; Bo et al., 2021), where individual word embeddings are perturbed with DP, with new words then sampled close to these privatized vectors. As Mattern et al. (2022b) point out, there are several shortcomings of such approaches, including a lack of obfuscating syntactic information and the inability to provide proper anonymization. In essence, these methods do not privatize a full utterance, but only single words.

3 Methods

We outline this section as follows. First, we briefly describe the baseline method we use, being a modified version of the ADePT system by Krishna et al. (2021). Next, we investigate two main issues with applying a local DP system such as ADePT to a transformer model, namely extreme sensitivity and computational infeasibility, described in Sections 3.2.1 and 3.2.2, respectively.

We then demonstrate several novel mechanisms which tackle these issues and provide numerous benefits in the privacy/utility trade-off for the local DP setting. Section 3.3 describes the clipping by value module, with an additional analysis on determining optimal settings for it provided in Appendix B. Sections 3.4 and 3.5 then describe the neuron-based pruning methods which significantly reduce the amount of noise that needs to be added to the model for a given privacy budget and increase model robustness to noise through further noisy training. Low-level specifics on the pruning methods are further provided in Appendix F.

3.1 Baseline (ADePT)

ADePT starts out with a standard autoencoder architecture. Given an input document x, an encoder function ENC calculates a latent vector representation z. This representation is then sent to a decoder function DEC, which reconstructs the original text $\hat{y}$. ADePT uses a single-layer, unidirectional LSTM for both the encoder and decoder.

$$z = \text{ENC}(x) \quad \text{and} \quad \hat{y} = \text{DEC}(z) \tag{1}$$

To incorporate differential privacy into this model, the unbounded latent vector $z \in \mathbb{R}^n$ (where n is the size of the autoencoder's hidden dimension) is bounded by its norm and the clipping constant $C \in \mathbb{R}$. Laplace or Gaussian noise (η) is then added to the resulting vector, from which the decoder reconstructs the original sequence, $\hat{y}$. For comparison with our primary methodologies below, we refer to this as the clipping by norm module, outlined in equation 2.

$$z' = z \cdot \min\left(1, \frac{C}{||z||_2}\right) + \eta \tag{2}$$

In our experiments, we make three adjustments to this system. First, we fix a theoretical issue in the sensitivity calculation for equation 2, outlined in Habernal (2021). Instead of using the sensitivity of $2C$ for the Laplace noise scale, outlined in Theorem 1 of Krishna et al. (2021), we instead use the corrected sensitivity of $2C\sqrt{n}$ from Theorem 5.1 of Habernal (2021). Second, the 'classical' Gaussian mechanism guarantees privacy only for $\varepsilon < 1$ (Dwork and Roth, 2013, p. 262). We therefore utilize the Analytic Gaussian mechanism (Balle and Wang, 2018) instead, which allows us to use $\varepsilon \geq 1$. Finally, we fix an issue with the pre-training procedure of the model. In Krishna et al. (2021), ADePT was pre-trained on the downstream datasets with clipping, but without the added DP noise from equation 2. Igamberdiev et al. (2022) demonstrated that this results in significant memorization by the model of the input documents, even after adding DP noise during the rewriting process. In order to remedy this, we therefore pre-train the autoencoder model on a public corpus, unrelated to the downstream datasets.

3.2 Applying LDP to Transformers

There are two main issues in applying a transformer model to a local DP setting similar to ADePT, outlined below.

3.2.1 Using LDP in pre-trained transformers suffers from extreme sensitivity

First, we need a significantly larger amount of noise to be added to the model, due to the increased size of the encoder output vector. Due to the cross-attention mechanism typical of transformer models, the full output vector for the BART encoder is of size $d_{tok} \times l$, where d_{tok} is the hidden size for a particular token, while l is the sequence length. For the smaller `bart-base` model, using a short sequence length of 20, this results in a dimensionality of $768 \times 20 = 15360$. In comparison, ADePT's encoder output vector dimensionality is only 1024 in our configuration.

3.2.2 High requirement of computational resources for pre-training

We experimented with clipping by norm for BART, similarly to ADePT, but found that it destroys any useful representations of the model (even prior to adding the DP noise). Additional pre-training of BART that would incorporate clipping by norm turned out to be ineffective.

The remaining option to learn a model with clipping by norm would be to pre-train the model from scratch. Unlike the small ADePT model, which is a unidirectional, single-layer LSTM, pre-training a BART transformer from scratch is computationally infeasible on an academic budget. While the details of BART's computational requirements are not described in Lewis et al. (2020), we can estimate this for the relatively small `bart-base` model of 139M parameters that was released by the original authors,[2] by comparison with other similar-sized models. For instance, the BERT model (Devlin et al., 2019), with less parameters (110M for `bert-base`), was pre-trained for 4 days on up to 16 TPUs, as described on the authors' Github repository.[3]

3.3 DP-BART-CLV (Clipping by Value)

To address the issues with clipping by norm, we developed the **DP-BART-CLV** model, shown in Figure 1. We analyzed the internal representations of a pre-trained BART model's encoder output vector values, using a public dataset. We found that these are mostly bounded within a couple of standard deviations from their mean. We present this analysis in detail in Appendix B.

To avoid significantly altering these representations, we can therefore use clipping by value (CLV), as in equation 3.

$$\bar{z}_i = \min(\max(z_i, C_{min}), C_{max}) \tag{3}$$

for any dimension i in the encoder output vector z, a set minimum threshold C_{min} and maximum threshold C_{max}. The bulk of values centered

[2] https://github.com/facebookresearch/fairseq/tree/main/examples/bart

[3] https://github.com/google-research/bert

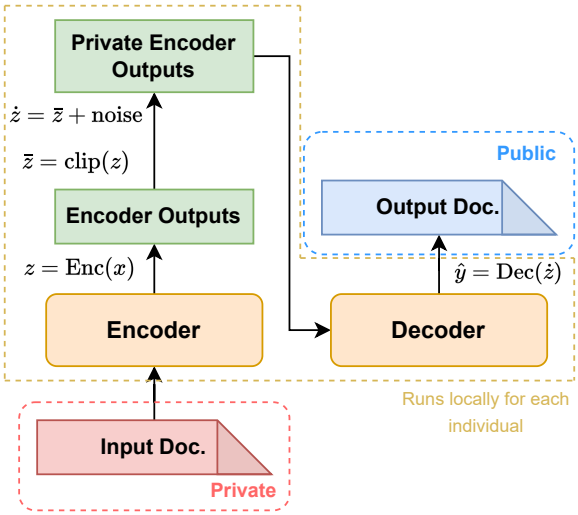

Figure 1: DP-BART-CLV

around the mean of z are thus left the same, without being rescaled as in equation 2. Since these values were also found to be symmetrically distributed, we modify equation 3 to set $C = C_{max} = -C_{min}$, as in equation 4.

$$\bar{z}_i = \min(\max(z_i, -C), C) \tag{4}$$

The pipeline for **DP-BART-CLV** is as follows. We first initialize a BART model using a pre-trained checkpoint, where pre-training was again done on a public dataset, separate from the downstream datasets that are to be privatized.

For a given document, we put it through the encoder of the model at inference time, obtaining the encoder output vector z, as in equation 5.

$$z = \text{ENC}(x) \tag{5}$$

where x is the input sequence and ENC is the encoder of the BART model. While the BART model outputs the encoder's last hidden state as $z \in \mathbb{R}^{l \times d_{tok}}$ for each mini-batch, we flatten this vector to be $z \in \mathbb{R}^n$, where $n = l \cdot d_{tok}$. Clipping is then performed as in equation 6,

$$\bar{z} = \text{CLIP}(z) \tag{6}$$

where CLIP is carried out for every dimension of the vector, according to equation 4.

With this clipping mechanism in place, we can now calculate its sensitivity, in order to determine the scale of noise to add in the DP setting. This is outlined in Theorems 3.1 and 3.2 below.

Theorem 3.1. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a function as in equation 6. The ℓ_1 sensitivity $\Delta_1 f$ of this function is calculated as in equation 7, where $C \in \mathbb{R} : C > 0$ is the clipping constant and $n \in \mathbb{N}$ is the dimensionality of the vector.*

$$\Delta_1 f = 2Cn \tag{7}$$

Proof. See Appendix C. □

Theorem 3.2. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a function as in equation 6. The ℓ_2 sensitivity $\Delta_2 f$ of this function is calculated as in equation 8, where $C \in \mathbb{R} : C > 0$ is the clipping constant and $n \in \mathbb{N}$ is the dimensionality of the vector.*

$$\Delta_2 f = 2C\sqrt{n} \tag{8}$$

Proof. See Appendix D. □

We then add noise to this clipped vector, as in equation 9.

$$\dot{z} = \bar{z} + (Y_1, \ldots, Y_n) \tag{9}$$

where each Y_i is drawn i.i.d. from $\text{Lap}(\frac{\Delta_1}{\varepsilon})$ for the Laplace mechanism (Dwork and Roth, 2013) or $\mathcal{N}(0, (\frac{\alpha\Delta_2}{\sqrt{2\varepsilon}})^2)$ for the Analytic Gaussian mechanism, where α is calculated according to Algorithm 1 of Balle and Wang (2018).

Decoding is then performed auto-regressively (e.g. using beam search), as usual, using this perturbed $\dot{z}$ encoder output vector, instead of the original z vector, as in equation 10.

$$\hat{y} = \text{DEC}(\dot{z}) \tag{10}$$

where $\hat{y}$ is the model's output prediction of the reconstructed input sequence x. By standard arguments, the **DP-BART-CLV** model satisfies $(\varepsilon, 0)$-DP for the Laplace mechanism and (ε, δ)-DP for the Analytic Gaussian mechanism, as outlined in equation 9 (Dwork and Roth, 2013; Balle and Wang, 2018).

3.4 DP-BART-PR (Pruning)

We develop the **DP-BART-PR** model in order to address the remaining issue of dimensionality, outlined in Section 3.2.1. The **DP-BART-CLV** model, while being resource-efficient, still has the issue of a large dimensionality for the encoder output vectors, since in equations 7 and 8, the sensitivity is multiplied by a factor of n and $\sqrt{n}$, respectively, which in turn results in a larger noise scale.

DP-BART-PR, addressing both the resource and dimensionality issues, is an extension to the above

DP-BART-CLV, with an additional iterative pruning/training mechanism applied to it. The procedure is outlined in Figure 2 and Algorithm 1 of Appendix E.

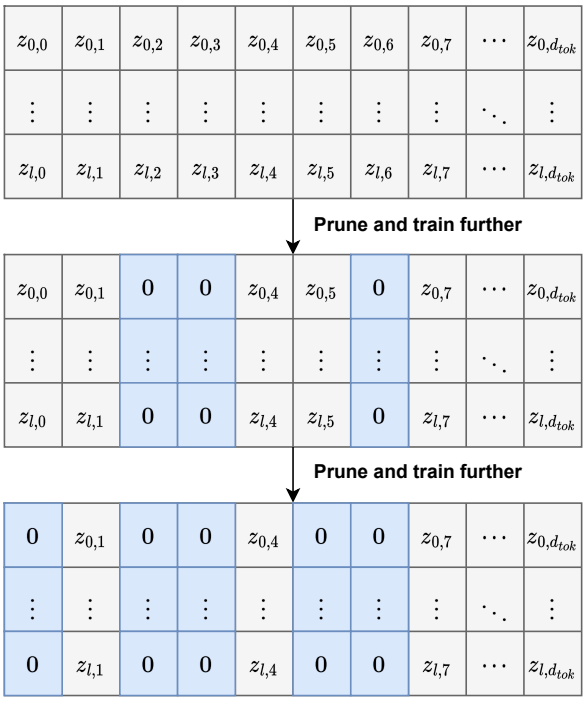

Figure 2: Pruning and re-training procedure for the DP-BART-PR model, illustrated for one document. Each i^{th} neuron from a set of indices is set to 0 for all tokens of the encoder output vectors $z \in \mathbb{R}^{l \times d_{tok}}$. These neuron indices are the same for any document. This process is repeated iteratively until performance starts to degrade.

As for **DP-BART-CLV**, we first load a pretrained BART model checkpoint. Each input token will have an encoder output representation of dimensionality d_{tok}. For every token in the sequence, we prune a certain percentage of these neurons by setting them to 0. Importantly, *these pruned neurons are the same for every single input document*. The criteria for selecting these pruned neurons is discussed in more detail in Appendix F.

Following this pruning step, we train the model for k iterations to compensate for possible lost performance from pruning. This step is performed on an external public dataset, unrelated to any downstream texts that are to be privatized. During this process, we also clip each dimension of the BART encoder output vector z_i according to equation 4, to encourage representations to be constrained within the ranges $-C$ and C to reduce potential negative performance impacts of clipping during the rewriting phase.

We note that only a few data points are necessary for this additional training step, maintaining the low-resource setting, outlined in Appendix I. We then continue this two-step process iteratively, until a desired dimensionality reduction of the encoder output vector is reached. At the end of this process, the resulting model weights are frozen and the final pruned indices of the encoder output vector z are saved. The model is then used for text rewriting at inference time, just like in **DP-BART-CLV**, but with the additional pruning step, using the saved indices.

As a result of this process, we can significantly reduce n in Equations 7 and 8, which in turn reduces the resulting noise scale used in equation 9. With less noise added to the encoder output vectors for any given ε value, we can thus expect a better privacy/utility trade-off.

This pruning procedure can thus be seen as a *privacy/utility tuning knob*. With more pruning, we reduce the size of n, therefore requiring less added noise for a given ε value in the DP setting. At the same time, more pruning reduces the model's expressivity with less dimensions, which will result in an inevitable performance drop after reaching a certain pruning threshold. We noticed that pruning a few dimensions (e.g. 25% of neurons) can recover basically all of the performance of the model with some additional training steps, but after a certain point this starts to degrade. The 'sweet spot' we found is at approximately 75% of neurons. Additional discussions on these points can be found in Appendix F. We would like to stress again that these pruning adjustments are made just once and using public data only, after which the final model can be used locally by any individual for their own data privatization.

3.4.1 Proof that DP-BART-PR is differentially private

Theorem 3.3. *The **DP-BART-PR** model, combining Algorithm 1 and the above **DP-BART-CLV** procedure, summarized in equation 9, satisfies $(\varepsilon, 0)$-DP when using the Laplace mechanism and (ε, δ)-DP when using the Analytic Gaussian mechanism.*

Proof. See Appendix G. □

3.5 DP-BART-PR+

We further augment the above **DP-BART-PR** model by incorporating additional training steps with added DP noise. This model follows the same procedure for iterative pruning and additional train-

ing, as outlined in algorithm 1, but we add further training iterations on the pruned model with added DP noise to the clipped encoder output representations, as in equation 9. For example, using the Analytic Gaussian mechanism at $\varepsilon = 500$, at each iteration we clip the encoder output vectors z from equation 5 and add the appropriate amount of Gaussian noise based on the sensitivity from equation 8.

The idea behind this additional training is to help the model to better decode from the noisified encoder representations. As with **DP-BART-PR**, for **DP-BART-PR+** we perform these additional training iterations on a public dataset, unrelated to the downstream datasets for privatized text rewriting. A separate model is prepared for each individual privacy budget ε.

4 Experiments

4.1 Datasets

We perform experiments on five English-language textual datasets, each gradually increasing in size (Table 1). For comparison with Krishna et al. (2021), we use ATIS (Dahl et al., 1994) and Snips (Coucke et al., 2018) as our 'small' datasets, with the task of multi-class intent classification. We use the same train/validation/test split as in Goo et al. (2018). For a medium-sized dataset, we use the popular IMDb dataset (Maas et al., 2011), on the binary classification task of movie review sentiment analysis. For this, as well as the following two datasets, we use a validation partition by randomly selecting 20% of the training set.

For a large dataset, we use the dataset from Gräßer et al. (2018), which is a collection of drug reviews from the website Drugs.com, also with the task of binary sentiment analysis as in Shiju and He (2022). This dataset, although publicly available, closely simulates a sensitive dataset in need of privacy protection, with detailed descriptions by users of their medical conditions and experiences with different treatments.

Our final dataset is the much larger Amazon Customer Reviews dataset (He and McAuley, 2016), of which we take a 2M subset of reviews from various categories (e.g. electronics, office products), from the full 144M. As with Drugs.com, we modify the original five-star sentiment score to a binary classification task, with four or more stars being the 'positive' class, while the rest are 'negative'. We refer to Appendix H for more details.

Dataset	Classes	# Trn.+Vld.	# Test
ATIS	26	4,978	893
Snips	7	13,774	700
IMDb	2	25,000	25,000
Drugs.com	2	161,297	53,766
Amazon	2	1,904,197	211,605

Table 1: Dataset statistics. Trn.: Train, Vld.: Validation. Size represents number of documents.

4.2 Experimental Setup

We have three main experimental configurations. The first is the **original** setting, where we run experiments on our downstream datasets without any rewriting or DP. The second configuration is **rewrite-no-dp**, where we utilize each of the four models outlined in Section 3 at $\varepsilon = \infty$ (**ADePT**, **DP-BART-CLV**, **DP-BART-PR**, **DP-BART-PR+**). Finally, the third and main configuration is **rewrite-dp**, where we compare the above four models, this time at various privacy settings ($\varepsilon \in [10, 2500]$, Laplace and Analytic Gaussian mechanisms).

For **rewrite-no-dp** and **rewrite-dp**, our experimental pipeline consists of the following four steps, depending on the specific model used:

Pre-training: The model is pre-trained on a large public corpus. For ADePT, we use 50% of the Openwebtext corpus (Gokaslan and Cohen, 2019). For all our BART experiments, we load a pre-trained `facebook/bart-base` model.[4]

Further training: Only for DP-BART-PR and DP-BART-PR+, again performed using the Openwebtext corpus. It helps the model adjust to pruning and DP noise, respectively (as outlined in Sections 3.4 and 3.5). More details on the amount of further training in Appendix I.

Rewriting: We take a pre-trained model and rewrite one of the downstream datasets.

Downstream: We take the rewritten dataset (training and validation partitions) and run downstream experiments on it using a pre-trained BERT model with a classification head on top. We use the rewritten validation set for hyperparameter optimization (see Appendix I) and the original test set for final evaluations. See Appendix J for details on the downstream model.

In the **original** setting, we use the same downstream model as above, using the original datasets

[4]Available from `https://huggingface.co/facebook/bart-base`

instead of the rewritten ones.

Evaluation We perform two types of evaluations for the above experimental settings: intrinsic and extrinsic. For our extrinsic evaluation we measure the test F_1 scores on the downstream task performance. This is the primary utility metric of the rewritten texts, with privacy correspondingly quantified with the ε value. We expect that even if a text may be rewritten to look very different from the original input, it could still have enough downstream task-specific information remaining to properly train a model on this task (e.g. the sentiment of a document in the case of sentiment analysis). This is in fact the 'sweet spot' we are looking for, removing identifying elements of the author, but still retaining some key features from the input for good downstream performance.

We also measure BLEU scores (Papineni et al., 2002) for our intrinsic evaluation, discussed in more detail in Appendix K.

5 Results

Figure 3 shows our downstream test F_1 results for all datasets, at varying values of ε. We report results for the Analytic Gaussian mechanism, which nearly always outperformed those of the Laplace mechanism. We present results in tabular form with mean and standard deviations in Appendix K. Additionally, we present sample rewritten texts in Appendix L. We outline the main patterns as follows.

DP-BART-PR+ performs best DP-BART-PR+ reaches the best privacy/utility trade-off for the majority of datasets, having the highest scores at the lower ε values. DP-BART-PR results are second-best for most datasets, performing better than DP-BART-CLV and ADePT, which are low for the majority of configurations. The overall results hierarchy can be clearly seen in the Snips dataset, where at $\varepsilon = 500$, DP-BART-PR+ reaches F_1 0.65, DP-BART-PR at 0.39, while both DP-BART-CLV and ADePT are below F_1 0.15.

Original vs. rewritten Results for the **original** setting are generally on-par with those of the **rewrite-no-dp** setting. For instance, Snips original F_1 is 0.98, and $\varepsilon = \infty$ with rewriting is also at F_1 of 0.98 for DP-BART-PR, being very similar for the other three models. One exception to this is IMDb, which has a drop from original F_1 0.86 to 0.72 for all models. This can be explained by the fact that the **original** settings use longer sequence lengths, while both **rewrite-no-dp** and **rewrite-dp** settings are limited to a sequence length of 20. This is not a problem for datasets such as ATIS and Snips, since their documents are generally very short, mostly limited to brief user inquiries. For a dataset such as IMDb, however, which consists of detailed reviews by individuals, limiting the sequence length results in a loss of valuable information.

Epsilon vs. dataset size Regardless of dataset size, we can see a drop in results for all models as ε is decreased. With the models incorporating pruning, this drop appears at later ε values, such as DP-BART-PR+ on the Amazon dataset moving down from F_1 0.82 at $\varepsilon = 250$ to F_1 0.33 at $\varepsilon = 100$, and DP-BART-PR from F_1 0.79 at $\varepsilon = 500$ to F_1 0.33 at $\varepsilon = 250$. A similar pattern can be seen for the Snips dataset, despite being far smaller than Amazon, while the Drugs.com dataset shows low results throughout, for all model types. The smallest dataset, ATIS, also performs poorly, which can be explained by the large number of classes and few data points for learning the task in the noisy setting. We can generally see that a larger dataset size does not necessarily mean better results at lower ε values, although the significantly larger Amazon dataset does show the best results.

6 Discussion

Reducing noise for text rewriting with LDP We have shown that it is possible to reduce the amount of noise in the LDP setting of privatized rewriting, in order to obtain more useful rewritten texts for downstream tasks. To compare DP-BART-CLV vs. DP-BART-PR, we can examine the resulting ℓ_2 sensitivity from equation 8 ($\Delta_2 f = 2C\sqrt{n}$). Setting sequence length $l = 20$ and $C = 0.1$, as in our experiments, without pruning we have a dimensionality of $n = 768 \cdot 20 = 15360$, hence $\Delta_2 f = 2 \cdot 0.1 \cdot \sqrt{15360} \approx 24.79$. With pruning we are able to remove 76.30% of those n neurons, with only $n = 182 \cdot 20 = 3640$ remaining. The ℓ_2 sensitivity thus becomes $\Delta_2 f = 2 \cdot 0.1 \cdot \sqrt{3640} \approx 12.07$.

Plugging this into the Analytic Gaussian mechanism's noise scale calculation from Balle and Wang (2018), with $\delta = 10^{-5}$ and $\varepsilon = 500$, we have $\sigma^2 = 0.8958$ without pruning and $\sigma^2 = 0.4362$ with pruning. We can therefore see that, **with DP-BART-PR, we are able to reduce the noise scale by more than half**.

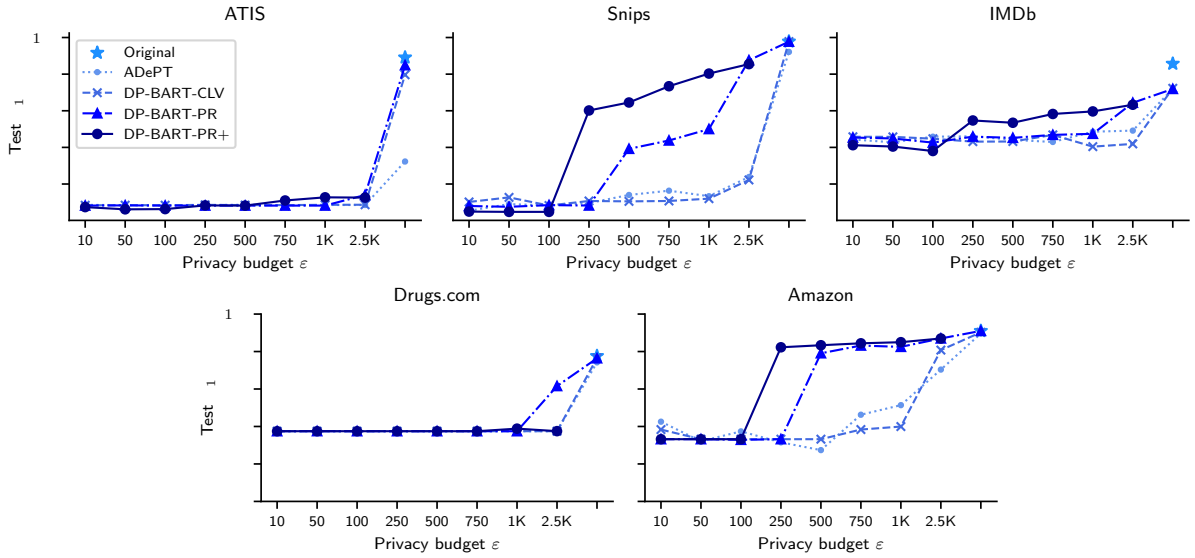

Figure 3: Downstream test F_1 results (macro-averaged) for each dataset, using the four model types. Lower ε corresponds to better privacy. Both **original** and **rewrite-no-dp** results can be seen on the right of each graph at $\varepsilon = \infty$. The rest of the results represent the **rewrite-dp** setting at different ε values.

Pre-training and computational resources Ultimately, a very effective way to prepare a model for privatized text rewriting would be to pre-train it from scratch, being fully in control of hyperparameters such as the dimensionality n of the encoder output vectors z, which determines the ℓ_1 and ℓ_2 sensitivities from equations 7 and 8, respectively. In addition, the whole model could be pre-trained with added noise and clipping mechanisms, potentially being even more robust than our approach in DP-BART-PR+, where we incorporate further noisy training. We noticed for DP-BART-PR+ that the lower the ε value we use, the more additional training iterations the model needs to properly reduce the validation loss.

This demonstrates that, also in the setting of pre-training from scratch, we would need to train for more iterations in order to reach lower ε values. This can pose serious challenges, however, for reasons of computational demand discussed in Section 3.2.2. DP-BART-PR+ can therefore be seen as a sweet spot approach, where we only need a few additional training iterations and can still achieve a significant dimensionality reduction through pruning, as well as additional robustness to noise.

What is being privatized It is very important to be clear on exactly what information is being privatized when performing text rewriting with LDP. Since we are working with DP at the document level, the entire document is a 'data point', hence any choice and combination of words for a given sequence would be a unique identifier. We thus avoid the problem of having to choose what specific tokens are 'private' within the document. This is crucial, since stylistic aspects of an author can be very abstract, with subtle syntactic and vocabulary choices.

Another significant benefit of such an approach, is that we are not limiting ourselves to any specific downstream analysis (e.g. sentiment of a document), being *task agnostic*. However, this also means that, for any given document, *any other document is neighboring*, since we are in the LDP setting. This leads us to a serious discussion on the limitations of such an approach in Section 8.

An additional question arises of whether one dataset may have multiple documents associated with one individual. There are several ways to go about dealing with this. One standard approach in differential privacy is to linearly scale the ε parameter. Thus, if there are k documents associated with a given individual, then a privacy budget of $k\varepsilon$ is accounted in total (Dwork and Roth, 2013). Another option would be to simply append all texts associated with one individual into a single 'document', rewriting this using just a single ε privacy budget.

7 Conclusion

We have proposed DP-BART, a novel methodology for LDP-based privatized text rewriting, which outperforms existing methods. We have demonstrated our method's privacy/utility trade-off, the relations between the privacy budget and dataset size, and discussed limitations of the privatized text rewriting approach as a whole. Future research directions include utilizing large-scale pre-training to potentially reach a better privacy/utility trade-off, as well as investigating domain specific text rewriting for relaxing the strict requirements of the LDP approach.

8 Limitations

Domain of public training texts In preparing the DP-BART models, it is important to take into account the domain of the public data that is used to (1) pre-train the original BART model, and (2) perform additional training iterations (DP-BART-PR and DP-BART-PR+). This will ultimately have an impact on the model's effectiveness for text privatization, depending on the nature of the downstream texts. For example, if this training data is restricted to news articles, then there may be limited performance for rewriting texts that are further from this domain, such as internet comments. Another obvious limitation is the language of the public data. If the model is trained on a monolingual English corpus, then it would not be possible to use it for rewriting texts from other languages.

The public data used for our experiments consists of news, web text, stories and books (Lewis et al., 2020; Gokaslan and Cohen, 2019). We expect that expanding this to include more data and more varied domains will lead to better performance in a greater diversity of texts and downstream tasks.

LDP for text rewriting For every output document, any two inputs, no matter how similar or distinct, are considered neighboring. If we have a small sequence length of 20 tokens, with a relatively small vocabulary of 1000 words, then the total number of possible combinations is 1000^{20}, which is 10^{60}! While we compress these documents into a latent vector with a limited range and dimensionality, the strict adjacency constraints are still present. We can therefore expect an inevitable utility drop when using more reasonable ε values (e.g. $\varepsilon = 1$).

With more sophisticated architectures, we have shown that it is possible to push this ε value down to some extent. However, our lowest ε is still too high to carry over into real-world applications of privacy preservation. As outlined by Hsu et al. (2014), values of ε for different applications in the DP literature can range from 0.01 to 10. Choosing the right ε value depends on the specific queries that are computed and the nature of the data (Lee and Clifton, 2011).

For our case, the value of ε can be interpreted in the following manner. The ε-LDP mechanism that we are applying to our data makes any two input texts rewritten to be indistinguishable up to a factor of e^{ε}. More formally, *for any two input texts* x and y to our LDP model $\mathcal{M}$:

$$\frac{\Pr[\mathcal{M}(x) = z]}{\Pr[\mathcal{M}(y) = z]} \leq e^{\varepsilon}, \tag{11}$$

where z is a given output text rewritten by the model.

This means that, when we set $\varepsilon = 250$, then any two texts will remain indistinguishable up to a factor of e^{250}. This is a very weak bound and, while it could provide some empirical privacy guarantees, on a theoretical level the privacy protection is not very strong. We can also see how this bound becomes exponentially stronger, as we decrease ε.

It may therefore make sense to take a slightly less strict approach to text adjacency, for instance moving into *domain specific* text rewriting. For example, text rewriting could be carried out for a specific dataset, with the notion of adjacency restricted to any two individuals within that dataset, hence requiring much less perturbation. The strength of the privacy guarantee, in this case, would then be very dependent on the size of the dataset (Mehner et al., 2021).

Acknowledgements

This project was supported by the National Research Center for Applied Cybersecurity ATHENE and by the PrivaLingo research grant (Hessisches Ministerium des Innern und für Sport). The independent research group TrustHLT is supported by the Hessian Ministry of Higher Education, Research, Science and the Arts. Thanks to Lena Held and Luke Bates for their helpful feedback and to Antti Honkela for very helpful hints regarding the limitations of the 'classical' Gaussian mechanism.

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.

Borja Balle and Yu-Xiang Wang. 2018. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. 2020. What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146.

Haohan Bo, Steven H. H. Ding, Benjamin C. M. Fung, and Farkhund Iqbal. 2021. ER-AE: Differentially private text generation for authorship anonymization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3997–4007, Online. Association for Computational Linguistics.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of bleu in machine translation research. In *11th conference of the european chapter of the association for computational linguistics*, pages 249–256.

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190*.

Deborah A Dahl, Madeleine Bates, Michael K Brown, William M Fisher, Kate Hunicke-Smith, David S Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the atis task: The atis-3 corpus. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006a. Our data, ourselves: Privacy via distributed noise generation. In *Annual international conference on the theory and applications of cryptographic techniques*, pages 486–503. Springer.

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

Cynthia Dwork and Aaron Roth. 2013. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407.

Oluwaseyi Feyisetan, Tom Diethe, and Thomas Drake. 2019. Leveraging hierarchical representations for preserving privacy and utility in text. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 210–219. IEEE.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Chih-Wen Goo, Guang Gao, Yun-Kai Hsu, Chih-Li Huo, Tsung-Chieh Chen, Keng-Wei Hsu, and Yun-Nung Chen. 2018. Slot-gated modeling for joint slot filling and intent prediction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 753–757.

Felix Gräßer, Surya Kallumadi, Hagen Malberg, and Sebastian Zaunseder. 2018. Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In *Proceedings of the 2018 International Conference on Digital Health*, pages 121–125.

Ivan Habernal. 2021. 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, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ivan Habernal. 2022. How reparametrization trick broke differentially-private text representation learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 771–777, Dublin, Ireland. Association for Computational Linguistics.

Song Han, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

Babak Hassibi, David G Stork, and Gregory J Wolff. 1993. Optimal brain surgeon and general network pruning. In *IEEE international conference on neural networks*, pages 293–299. IEEE.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th international conference on world wide web*, pages 507–517.

Justin Hsu, Marco Gaboardi, Andreas Haeberlen, Sanjeev Khanna, Arjun Narayan, Benjamin C Pierce, and Aaron Roth. 2014. Differential privacy: An economic method for choosing epsilon. In *2014 IEEE 27th Computer Security Foundations Symposium*, pages 398–410. IEEE.

Timour Igamberdiev, Thomas Arnold, and Ivan Habernal. 2022. DP-rewrite: Towards reproducibility and transparency in differentially private text rewriting. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 2927–2933, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Timour Igamberdiev and Ivan Habernal. 2022. Privacy-Preserving Graph Convolutional Networks for Text Classification. In *Proceedings of the Language Resources and Evaluation Conference*, pages 338–350, Marseille, France. European Language Resources Association.

Diederik P Kingma and Max Welling. 2014. Auto-encoding variational bayes. *2nd International Conference on Learning Representations, ICLR*.

Oleksandra Klymenko, Stephen Meisenbacher, and Florian Matthes. 2022. Differential privacy in natural language processing the story so far. In *Proceedings of the Fourth Workshop on Privacy in Natural Language Processing*, pages 1–11, Seattle, United States. Association for Computational Linguistics.

Satyapriya Krishna, Rahul Gupta, and Christophe Dupuy. 2021. 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, Online. Association for Computational Linguistics.

John K Kruschke and Javier R Movellan. 1991. Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks. *IEEE Transactions on systems, Man, and Cybernetics*, 21(1):273–280.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Yann LeCun, John Denker, and Sara Solla. 1989. Optimal brain damage. *Advances in neural information processing systems*, 2.

Jaewoo Lee and Chris Clifton. 2011. How much is enough? choosing ε for differential privacy. In *International Conference on Information Security*, pages 325–340. Springer.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 142–150.

Justus Mattern, Zhijing Jin, Benjamin Weggenmann, Bernhard Schoelkopf, and Mrinmaya Sachan. 2022a. Differentially private language models for secure data sharing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4860–4873, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Justus Mattern, Benjamin Weggenmann, and Florian Kerschbaum. 2022b. The limits of word level differential privacy. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 867–881.

Luise Mehner, Saskia Nuñez von Voigt, and Florian Tschorsch. 2021. Towards explaining epsilon: A worst-case study of differential privacy risks. In *2021 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 328–331. IEEE.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318. PMLR.

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

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Manuel Senge, Timour Igamberdiev, and Ivan Habernal. 2022. One size does not fit all: Investigating strategies for differentially-private learning across NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, UAE.

Akhil Shiju and Zhe He. 2022. Classifying drug ratings using user reviews with transformer-based language models. In *2022 IEEE 10th International Conference on Healthcare Informatics (ICHI)*, pages 163–169.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Teng Wang, Xuefeng Zhang, Jingyu Feng, and Xinyu Yang. 2020. A comprehensive survey on local differential privacy toward data statistics and analysis. *Sensors*, 20(24):7030.

Benjamin Weggenmann, Valentin Rublack, Michael Andrejczuk, Justus Mattern, and Florian Kerschbaum. 2022. DP-VAE: Human-Readable Text Anonymization for Online Reviews with Differentially Private Variational Autoencoders. In *Proceedings of the ACM Web Conference 2022*, pages 721–731, Virtual Event. ACM.

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, Online. Association for Computational Linguistics.

A Background

Differential Privacy Differential privacy (DP), originally proposed by Dwork et al. (2006b), is a formal guarantee that the output of some analysis on a given dataset is nearly indistinguishable when one data point is modified. In other words, no individual can stand out as a result of this analysis, preserving their privacy.

To define this more formally, we first outline the notion of *neighboring datasets*.

Definition A.1. *Two datasets D and D' are considered **neighboring** if they differ in at most one record, i.e., one individual's data point. This means that either $D' = D \pm 1$, or $D' = D$ with the i-th data point replaced.*

In DP, we typically refer to a *query* on a dataset, as defined below.

Definition A.2. *A **query** is a function $f : D \to \mathbb{R}^k$ that we evaluate on a dataset D.*

This can range from simpler queries, such as taking the average length of a document, to more complex queries, e.g. a deep learning model predicting the sentiment of a document.

In order to provide a formal privacy guarantee, we add *randomness* to this query by perturbing $f(D)$. We refer to this randomized function as a *randomized mechanism* $\mathcal{M}(D; f)$.

The formal definition of differential privacy can now be described as follows.

Definition A.3. *For $\varepsilon \geq 0$ and $\delta \in [0, 1]$, a mechanism $\mathcal{M}$ is (ε, δ)-differentially private if, for all $S \subseteq$ Range$(\mathcal{M})$, and for any two neighboring datasets D and D', the following holds true:*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D') \in S] + \delta \quad (12)$$

Importantly, ε is the *privacy budget*. The lower it is, the more private the mechanism is, since the two output distributions of $\mathcal{M}(D)$ and $\mathcal{M}(D')$ are constrained to be more similar. While the original definition of DP only included this term, the additional additive term δ was later introduced in Dwork et al. (2006a) and represents the 'cryptographically small' probability that pure ε-DP is broken. In the case where $\delta = 0$, we return to the original, or *pure differential privacy* setting.

In order to make a query differentially private, random noise is added based on the query's *sensitivity*, or the maximum amount that the output of the query can change. This represents the degree to which one individual can affect f in the worst case. In turn, this is also the amount of noise that has to be introduced to f in order to obscure one individual's contribution.

Finally, there are two primary settings for differential privacy: **global DP** and **local DP (LDP)**, depicted in Figure 4. In the former case, the query $f(D)$ is first evaluated, and then perturbed by a trusted data aggregator. In contrast, in LDP *each individual data holder perturbs his/her own data point*, prior to data collection and without replying on a third-party. As noted by Wang et al. (2020), in LDP *any two data points* are considered neighboring, in contrast to the global DP definition of two datasets differing in one record.

Since each individual is fully in control of the privatization process, this makes LDP a particularly attractive setting for providing a privacy guarantee. The difficulty, however, is that we typically require orders of magnitude more perturbation for f than we otherwise would need in the global DP setting. Refer to Igamberdiev and Habernal (2022); Senge et al. (2022); Habernal (2022) for further use-case examples in NLP.

Overview of the BART Model The BART model is a sequence-to-sequence Transformer ar-

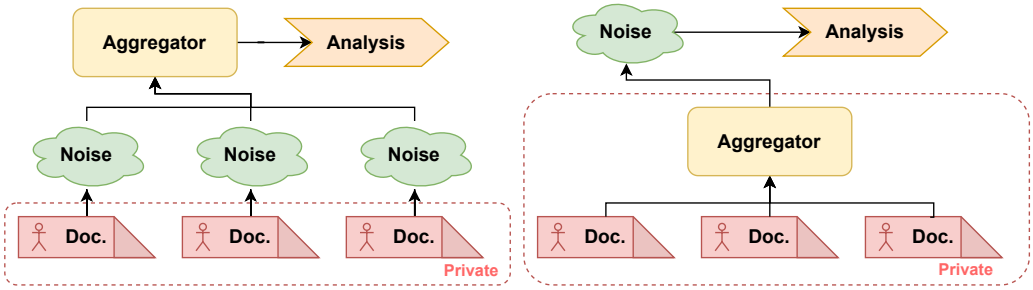

Figure 4: Local DP (left) vs. global DP (right). In the local framework, the aggregator does not have access to the original data, with each individual applying DP to their own private data point. In the global framework, the aggregator adds DP noise to the original data, given a specific query from an analyst.

chitecture (Vaswani et al., 2017), acting as a denoising autoencoder. It combines the BERT-like bidirectional encoder (Devlin et al., 2019) with the GPT-like left-to-right autoregressive decoder (Radford et al., 2018). The base model contains 6 layers for the encoder and decoder, with cross-attention performed over the final encoder layer. BART is pre-trained through a number of noise transformations of the input document, including token masking, token deletion, and sentence permutation, optimizing a cross-entropy reconstruction loss. One strong benefit of BART for differentially private text rewriting is that, by design, it is well-equipped for the autoencoding task of reconstructing corrupted documents.

Overview of pruning for neural networks The more popular technique is weight pruning (e.g. LeCun et al. (1989), Hassibi et al. (1993), Frankle and Carbin (2018)), reducing the size of a model and its computation time, but minimizing any negative impact to its performance. More distinct is structured pruning, such as neuron pruning (e.g. Kruschke and Movellan (1991)), where the architecture of a network is reduced by eliminating full structures of the network, which is more in line with our approach. Regardless of the specific method used, a very common pipeline is to iteratively prune and further train a model, to help it recover from potentially lost performance (Han et al., 2015). Overall, pruning tends to be highly effective, with substantial compression possible for models (Blalock et al., 2020).

In contrast to the above goals of size and computational efficiency, we use pruning with the primary objective of *dimensionality reduction* on a specific hidden layer. This dimensionality is directly related to privacy concerns, with a lower dimension resulting in less added noise to the model, which allows us to use lower privacy budgets while maintaining better performance. Our neuron-based pruning approach is outlined in Section 3.4.

B Selecting Clipping Value for DP-BART

When clipping encoder outputs by value for the DP-BART model, we want to choose left and right values C_{min} and C_{max} that capture the most information from the original vector. One way to go about this, is to estimate the distribution of the encoder output vectors $z \in \mathbb{R}^n$ (see equation 5) of a pre-trained BART model checkpoint, given several documents from an external public dataset, and then clip a certain number of standard deviations from the estimated mean. Performing an exploratory data analysis on these encoder output vectors, we noticed that they fairly closely match a Gaussian distribution, although with far more outliers.

In order to look into this more closely, we can perform Maximum Likelihood Estimation (MLE) to estimate the μ and σ^2 parameters, assuming the data follows a Gaussian distribution. For Gaussians, the MLE of these two parameters is simply the mean and variance of the existing data, respectively, in our case of the values of z, given an input document x. Based on multiple documents, we find that $\mu \approx 0.00$ and $\sigma \approx 0.2$. Hence, using the *66-95-99.7 rule* for normal distributions, we can clip two standard deviations to the left and right and retain 95% of the original values.

We therefore initially set $C = C_{max} = -C_{min}$, where $C = \mu + \sigma \cdot 2 = 0 + 0.2 \cdot 2 = 0.4$. Since μ is found to be 0, we are able to simplify the calculation to only have the C and σ parameters. In practice, we found that clipping only half of one standard deviation was enough to retain good performance, despite clipping away more informa-

tion than what we estimate above. Hence, we set $C = \sigma/2 = 0.1$.

C Proof of Theorem 3.1

Proof. The ℓ_1 sensitivity of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is defined as: $\Delta_1 f = \max_{x,y} ||f(x) - f(y)||_1$, where $||x-y||_1 = 1$. Since in our case f clips every value to be in the range $[-C, C]$, the following inequality must be true.

$$\begin{aligned} ||f(x) - f(y)||_1 &= |f(x_1) - f(y_1)| + \dots \\ &\quad + |f(x_n) - f(y_n)| \\ &\leq |C - (-C)| + \dots \\ &\quad + |C - (-C)| \\ &= |2C| + \dots + |2C| \\ &= 2Cn \end{aligned} \tag{13}$$

This inequality also holds true when the C values are reversed for any summand, due to the absolute value: $|C - (-C)| = |-C - C|$.

□

D Proof of Theorem 3.2

Proof. The ℓ_2 sensitivity of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is defined as: $\Delta_2 f = \max_{x,y} ||f(x) - f(y)||_2$, where $||x-y||_1 = 1$. As for the ℓ_1 sensitivity above, f clips every value to be in the range $[-C, C]$, so the following inequality must be true.

$$\begin{aligned} ||f(x) - f(y)||_2 &= \sqrt{\begin{aligned}&(f(x_1) - f(y_1))^2 + \dots \\ &\quad + (f(x_n) - f(y_n))^2\end{aligned}} \\ &\leq \sqrt{\begin{aligned}&(C - (-C))^2 + \dots \\ &\quad + (C - (-C))^2\end{aligned}} \\ &= \sqrt{(2C)^2 + \dots + (2C)^2} \\ &= 2C\sqrt{n} \end{aligned} \tag{14}$$

This inequality also holds true when we reverse the position of the C values for any summand, $(C - (-C))^2 = (-C - C)^2$.

□

E Pruning algorithm for DP-BART-PR

We present the procedure for pruning neurons in DP-BART-PR in Algorithm 1.

Algorithm 1 DP-BART Pruning

Input: Encoder: ENC_{θ_0}, Decoder: DEC_{θ_0}, Public dataset: $\mathcal{D}$, Encoder output dimension per token: d_{tok}, Number of epochs to additionally train: E

Output: Pruned model: ENC_{θ_E}, DEC_{θ_E}; Array of neuron indices to prune P of size d_{tok}

function PRUNE(z, P)
  ▷ $z \in \mathbb{R}^{l \times d_{tok}}$, where l is the seq. length
  for j in 1 to d_{tok} **do**
    if j in P **then**
      ▷ Set that neuron to 0 for all tokens
      $z[:, j] \leftarrow 0$
  return z
function ITER_PR($\mathcal{D}$, ENC_θ, DEC_θ, P)
  ▷ Iterate with pruning
  for each document x in $\mathcal{D}$ **do**
    Compute encoder outputs, $z \leftarrow \text{ENC}_\theta(x)$
    Prune, $z_{pr} \leftarrow$ PRUNE(z, P)
    Decode, $\hat{y} \leftarrow \text{DEC}_\theta(z_{pr})$
    Compute loss on $\hat{y}$ and optimize
function ADD_P_IDXS(P)
  $new\_idxs \leftarrow$ select k values in $[1, d_{tok})$
  Append new_idxs to P
  return P

$P \leftarrow$ new Array
for epoch e in 1 to E **do**
  $P \leftarrow$ ADD_P_IDXS(P)
  ITER_PR($\mathcal{D}$, ENC_{θ_e}, DEC_{θ_e}, P)
return ENC_{θ_E}, DEC_{θ_E}, P

F Selecting Neurons for Pruning

At each pruning/training iteration for preparing the DP-BART-PR model, we need some criteria for selecting the next set of neuron indices that will be set to 0. Our method for selecting these is generally in line with previous work on pruning (Blalock et al., 2020), using weight magnitudes to determine relative importance of those weights.

In the cross-attention module of the decoder of a transformer model such as BART, there are three initial projections of the input or target representations: Key (K), Query (Q), and Value (V). The K and V projections come directly from the encoder output vectors multiplied by a weight matrix for each, while the Q projection comes from the decoder's intermediate representations multiplied by a weight matrix. We can therefore choose the

weight matrix of either the K or V projection from the cross-attention module of one of the decoder's layers. For this weight matrix, we take the sum of absolute values of all weights associated with a particular neuron, to give a general indication of its importance. Given the distribution of these values associated with each neuron, we take the 25% quantile as the threshold. Any neuron with a value below this threshold is selected for pruning and set to 0.

At the next pruning iteration, after further training, we repeat the above process, this time only taking into account neurons that have not already been set to 0. We again calculate each neuron's relative importance value, taking the 25% quantile of these new values as the next threshold, and selecting any neurons with an associated importance value below it for pruning.

We found that taking the weight matrix of the K projection from the initial decoder layer outperformed all other configurations, such as using subsequent layers, or the V projection. Additionally, we found the above method to outperform randomly pruning neurons.

We perform two additional tweaks to this process to improve results further. First, we include the clipping by value procedure, with $C = 0.2$, when further training the model at each pruning iteration. We found that, without this step, the encoder output representations tend to shift to a distribution of values with a greater standard deviation. This then requires a larger C value when determining the mechanism's sensitivity in equations 7 and 8, which in turn requires a greater noise scale in equation 9. By including this clipping, we encourage encoder output representations to continue to primarily stay within the range $(-C, C)$.

The other tweak that we found to further improve results is to prune and further train the BART model for k iterations, but then use the neuron indices for pruning from the $k-1$ iteration. Performing this full pruning pipeline on a public dataset, we found that the best BLEU scores for rewriting at various ε values are after pruning/training the model for 6 iterations, then using the pruning indices from the 5^{th} iteration for actual rewriting of downstream datasets. This amounts to a total of 586 out of 768 (76.30%) neurons pruned for each token.

In theory, this pruning procedure could be replaced with another dimensionality reduction technique for the last hidden state of the encoder outputs (e.g. a bottleneck layer and its inverse). In our experiments, however, the above pruning procedure produced superior results when trying various options for such a bottleneck layer. This includes architectures such as a feedforward neural network and CNN (LeCun et al., 1998), as well as various training methods (e.g. training these layers separately and reinserting them into the final full model, or training the full model together with these layers).

G Proof of Theorem 3.3

Proof. The procedure outlined in Algorithm 1 is performed on a public dataset, unrelated to the downstream data that is considered sensitive, hence no privacy budget is used up.

The remaining rewriting procedure with the pruned indices is exactly the same as for **DP-BART-CLV**, just at a lower dimension. The neuron indices that are set to 0 are the same for any input document. This means that no information from the input is encoded in these neuron indices. From the DP point of view, these zeroed neurons are the same for any two neighboring data points. Therefore, these neurons have no contribution to the DP sensitivity and do not require any privatization. The same proofs are therefore valid as for Theorems 3.1 and 3.2 for the Laplace and Analytic Gaussian mechanisms, respectively. □

H Preparation of Larger Datasets

H.1 Drugs.com reviews dataset

We present additional statistics on the Drugs.com dataset in Table 2. We note the class imbalance of the original dataset, where the majority class was the highest rating 9, from a score of 0 to 9, which accounted for approximately 17% of the total training set. This contributes to the relative imbalance of the positive and negative classes in our binary class version of the dataset.

	# Train	# Test
# Positive	97,410	32,349
# Negative	63,887	21,417
# Total	161,297	53,766

Table 2: Class distributions and total documents for the Drugs.com reviews dataset. Original classes 8 and 9 converted to the *positive* class, while the rest to the *negative* class for our experiments.

H.2 Amazon reviews dataset

For the Amazon dataset, since using the full 144M reviews is too computationally expensive, we reduce this to a more practical size, while still being comparatively larger than the other downstream datasets. To prepare a subset of the full Amazon dataset, we first select several product categories based on four criteria. (1) The category is large enough (e.g. $> 2M$ reviews). (2) Label 5 for the star rating is not too dominant (e.g. $< 60\%$), see general imbalance outlined in Table 3. (3) Label 4 for the star rating is also not too dominant (e.g. $< 60\%$), since we are merging labels 5 and 4 into the *positive* class. (4) Label 1 for the star rating has enough representation (e.g. $> 10\%$).

We selected a total of 7 product categories, which matched at least three out of four of these criteria. From these reviews, we then filtered to include only those with 20 tokens or less, to fit our experimental scenario of shorter documents (outlined in more detail in Appendix I). We then reduced this further by balancing positive and negative classes, with uniform probability selecting only N_{neg} positive label reviews, where N_{neg} is the number of negative labels in our current subset. Finally, we uniformly selected two-thirds of the resulting balanced dataset to reach the final size of approximately 2M reviews. We present each product category and its corresponding size in Table 3.

Importantly, we have a well-defined train-test split, taking 10% of the processed dataset and setting it aside for final downstream test evaluations. We release the specific document indices of our subset from the original large Amazon reviews dataset.[5] We present the final dataset statistics in Table 4.

I Hyperparameter Configuration

For all our model configurations, we use a sequence length of 20 tokens. This limits the sensitivity in equations 7 and 8 for our three BART models. For the ADePT model, we found that it is generally ineffective at the autoencoding task when using larger sequence lengths, presumably due to the problem of vanishing gradients for RNN-based models (Pascanu et al., 2013). Our search space for learning rates is in the range $[10^{-6}, 0.01]$. We use batch sizes of either 32 or 64.

When pre-training ADePT, we include the clipping procedure from equation 2, otherwise the model is unable to properly rewrite a given input document, since the clipping significantly alters the encoder output representations. Additional hyperparameters for ADePT include an embedding size of 300 with pre-trained GloVe embeddings[6] (Pennington et al., 2014) and a hidden size of 512. Combining the LSTM cell and hidden state sizes, the ADePT encoder output vectors have a dimensionality of $512 \cdot 2 = 1024$.

For rewriting using the Analytic Gaussian mechanism, we always keep the δ value below $1/N$, where N is the total number of documents for a given dataset. This is based on the idea that using a δ value that is overly large in relation to the dataset size can lead to potential privacy leaks, hence maintaining $\delta \ll 1/N$ is a good guideline to follow (Abadi et al., 2016). We therefore use a δ value of 10^{-5} for the ATIS, Snips and IMDb datasets, 10^{-6} for the Drugs.com dataset, and 10^{-7} for Amazon reviews. We perform rewriting with beam search, using a beam size of 10.

When performing additional training for the DP-BART-PR model, we again use the Openwebtext corpus. At each stage of pruning, we train the model for 500 iterations at a batch size of 32. In the case of further training for the DP-BART-PR+ model, we again use the Openwebtext corpus, with the same number of iterations and batch size, but performed over multiple epochs. The number of epochs ranges from 100 to 500, for the different ε values from 2500 down to 10, based on the prediction loss and intermediate model outputs. We applied these further training steps to the DP-BART-CLV model as well to account for the potential effects of this training alone, but we did not find any improvements. This is in line with the high dimensionality issue of DP-BART-CLV destroying input representations in the private setting, which this additional training does not resolve without the pruning adjustments of the DP-BART-PR(+) models.

Regarding downstream text classification experiments, we run each configuration for a maximum of 50 epochs with three random seeds and report the mean. We use an early stopping patience of 5

[5] Original full dataset available on Huggingface at https://huggingface.co/datasets/amazon_us_reviews, our subset available at https://github.com/trusthlt/dp-bart-private-rewriting/tree/main/assets/amazon_reviews_subset.

[6] Downloaded from https://nlp.stanford.edu/data/glove.6B.zip

Product Cat.	# Docs. (original)	# Docs. (subset)
Digital_Video_Games_v1_00	145,341	11,375
Electronics_v1_00	3,093,869	201,708
Lawn_and_Garden_v1_00	2,557,288	202,226
Major_Appliances_v1_00	96,901	4,940
Mobile_Apps_v1_00	5,033,376	536,550
Office_Products_v1_00	2,642,434	182,202
Wireless_v1_00	9,002,021	976,801
Total	22,571,320	2,115,802

Table 3: Product categories and corresponding number of documents from the full Amazon reviews dataset (mid), as well as from our prepared subset (right).

	# Train	# Test
# Positive	952,153	105,797
# Negative	952,044	105,808
# Total	1,904,197	211,605

Table 4: Final class distributions and total reviews for our Amazon reviews subset.

epochs. We also report the standard deviation in Appendix K. We outline our choice of the clipping by value constant C in Appendix B and amount of pruning in Appendix F.

Finally, our computational runtimes are under 1 hour for each configuration that does not use the Amazon dataset. The only exception to this is the Drugs.com reviews dataset, which reaches up to 2 hours 10 minutes for rewriting with the DP-BART models. The Amazon dataset takes significantly longer, with approximately 24 hours for rewriting with ADePT, 47 hours rewriting with DP-BART models, as well as up to 18 hours for downstream experiments, depending on when the early stopping condition is reached. We run experiments on a 32GB NVIDIA V100 Tensor Core GPU.

J Downstream Experimental Setup

We use a pre-trained BERT model (Devlin et al., 2019) for running downstream experiments on the rewritten texts. We add a feedforward layer on top of the BERT model, taking as input the mean of its last hidden states. The model predicts the output label for text classification. For training the model and running validation, we use the rewritten training and validation partitions for each downstream dataset, at a given privacy configuration. For final evaluation, we run the model on the original test set of each dataset.

K Intrinsic evaluations and detailed downstream results

For intrinsic evaluation, we use BLEU scores to measure how close the input and rewritten output texts are to one another. Despite some criticisms of BLEU as a general-purpose evaluation metric for text generation (e.g. Callison-Burch et al. (2006)), it perfectly fits our scenario. Being a metric of n-gram overlap, it allows us to compare how similar the inputs and outputs are. In a way, a very high BLEU score points to privacy leakage, since it is showing how much of the original text remains in the output. We would therefore expect well privatized texts to have a relatively low BLEU score.

Our results can be seen in Table 5 for rewriting the training partition of each dataset with the Analytic Gaussian mechanism, together with the detailed downstream test F_1 results.

We can see that the BLEU scores for the training partition of each dataset show a largely positive correlation with the test F_1 downstream results, where a decrease in the former also indicates a decrease in the latter. For instance, the Snips dataset shows a BLEU score of 0.31 at $\varepsilon = 2500$ for DP-BART-PR+, with a test F_1 score of 85%. At $\varepsilon = 750$, this drops down to 0.23 BLEU score and 73% test F_1. By $\varepsilon = 250$, the BLEU score is at 0.07, with the test F_1 score at 60%. Interestingly, despite lower BLEU scores, the downstream model is still able to sometimes learn the task successfully, obtaining a good F_1 score on the original test set.

Another example of this can be seen for the DP-BART-PR model on the Amazon dataset at $\varepsilon = 1000$, with a BLEU score of 0.17, reaching a test F_1 of 82%. A similar instance is DP-BART-PR+ rewriting Amazon at $\varepsilon = 250$, with a BLEU score of 0.15 and a test F_1 of 82%, compared to the non-private F_1 of 91%. This is in line with

the goals of text privatization, where original identifying elements of the text are removed, but key features from the input are retained for good downstream performance.

L Sample rewritten texts

L.1 Comparing rewritten texts across privacy budgets

Original It slows the game performance a bit, but it's totally worth it!
$\varepsilon = \mathbf{2500}$ The performance of the game is a bit sluggish, but it's worth it
$\varepsilon = \mathbf{1000}$ It's that time of year again. But if you slow down your
$\varepsilon = \mathbf{750}$ It's that time of year again when we talk about kitty racing
$\varepsilon = \mathbf{500}$ We've all been talking about the game, but this is a bit of
$\varepsilon = \mathbf{250}$ 12 years ago today morning morning morning, a 12-
Sample rewritten texts for varying privacy budgets, using DP-BART-PR+ for the Amazon dataset.

Original i want to hear something eclectic
$\varepsilon = \mathbf{2500}$ The following is a list of interesting things to hear from the eclectic, eclectic, and
$\varepsilon = \mathbf{1000}$ i want to hear something different from what everyone else has been hearing about this week.
$\varepsilon = \mathbf{750}$ i want to hear something different about this mod. It's simple, but
$\varepsilon = \mathbf{500}$ i want to hear something like this. If you want to listen to music
$\varepsilon = \mathbf{250}$ In the last three year in the last time it seems to have an area of the
Sample rewritten texts for varying privacy budgets, using DP-BART-PR+ for the Snips dataset.

We provide sample rewritten texts from the DP-BART-PR+ model, comparing the difference in output across ε values on the Snips and Amazon datasets. We can see that, for different values of ε, parts of the original input sequence reappear in the rewritten output to varying degrees. For example, the first five tokens of the original Snips sample reappear in the rewritten texts at $\varepsilon = 500, 750, 1000$. At the lower ε value of 250, while the output is still in part coherent, it is no longer recognizable from the original. At the lowest ε values, there is so much noise added to the model that the output primarily consists of 'start of sequence' and 'end of sequence' tokens, resulting in an overall empty output. For the Amazon example, most rewritten tokens are different from the input, with some resemblance at $\varepsilon = 500$, but a more coherent and related output primarily at the larger $\varepsilon = 2500$.

Interestingly for these examples, while the rewritten documents are very altered from the original documents throughout, it is enough in the case of DP-BART-PR+ to achieve a relatively good downstream performance, such as an F_1 score of 0.65 for Snips at $\varepsilon = 500$ and 0.82 for Amazon at $\varepsilon = 250$. This is more of what we would expect from a text rewriting system, since if the original text is clearly noticeable in the rewritten output, we would strongly suspect a privacy leak.

L.2 Comparing rewritten texts across models

Original The product doesn't work at all.
ADePT has ! low phone unauthorised and 1 awesome 5th whatsoever pickle my canna kindle just flowed phones signup
DP-BART-CLV """". @...???)!).. W @. W???)
DP-BART-PR Technical precisely anticipate work-touch to enhance Resources Resources ARE/and and Science Matters/
DP-BART-PR+ "The product doesn't work at all." That is the sentiment of
Sample rewritten texts for each model type, at $\varepsilon = 750$ for the Amazon dataset.

We additionally provide sample rewritten texts from each model, at the same ε value and on the same dataset (Amazon at $\varepsilon = 750$). Here we can see that the DP-BART-PR+ model output is the most similar to the original document, being rewritten verbatim, followed by some additional output. The output sequence for DP-BART-PR is less coherent, but still with recognizable sequences for some token pairs, while DP-BART-CLV and ADePT have output that is seemingly random.

Dataset	ε	Original	ADePT		DP-BART-CLV		DP-BART-PR		DP-BART-PR+	
		Test F_1	BLEU	Test F_1	BLEU	Test F_1	BLEU	Test F_1	BLEU	Test F_1
Snips	∞	0.98 (0.00)	6.34	0.92 (0.00)	98.41	0.98 (0.00)	54.39	0.98 (0.00)	N/A	N/A
	$2,500$		0.16	0.24 (0.13)	0.02	0.22 (0.14)	2.19	0.88 (0.03)	0.31	0.85 (0.02)
	$1,000$		0.03	0.13 (0.09)	0.00	0.12 (0.10)	0.07	0.50 (0.07)	0.30	0.80 (0.02)
	750		0.02	0.16 (0.08)	0.00	0.11 (0.07)	0.02	0.44 (0.11)	0.23	0.73 (0.04)
	500		0.01	0.14 (0.08)	0.00	0.11 (0.06)	0.01	0.39 (0.10)	0.22	0.65 (0.01)
	250		0.01	0.10 (0.09)	0.00	0.11 (0.07)	0.00	0.08 (0.02)	0.07	0.60 (0.03)
	100		0.01	0.08 (0.02)	0.00	0.08 (0.02)	0.00	0.08 (0.02)	0.00	0.05 (0.02)
	50		0.01	0.09 (0.05)	0.00	0.13 (0.06)	0.00	0.08 (0.03)	0.00	0.05 (0.02)
	10		0.01	0.05 (0.01)	0.00	0.10 (0.04)	0.00	0.08 (0.03)	0.00	0.05 (0.01)
ATIS	∞	0.89 (0.01)	16.04	0.32 (0.01)	97.45	0.80 (0.03)	69.26	0.85 (0.01)	N/A	N/A
	$2,500$		0.45	0.09 (0.00)	0.02	0.09 (0.00)	2.13	0.14 (0.07)	0.24	0.13 (0.07)
	$1,000$		0.06	0.09 (0.00)	0.01	0.09 (0.00)	0.06	0.08 (0.00)	0.25	0.13 (0.03)
	750		0.05	0.08 (0.00)	0.00	0.08 (0.00)	0.03	0.08 (0.00)	0.24	0.11 (0.05)
	500		0.03	0.09 (0.00)	0.00	0.08 (0.00)	0.01	0.08 (0.00)	0.11	0.08 (0.00)
	250		0.01	0.08 (0.00)	0.00	0.09 (0.00)	0.01	0.08 (0.00)	0.08	0.08 (0.00)
	100		0.01	0.08 (0.00)	0.00	0.08 (0.00)	0.00	0.08 (0.00)	0.00	0.06 (0.04)
	50		0.01	0.08 (0.00)	0.00	0.08 (0.00)	0.00	0.08 (0.00)	0.00	0.06 (0.04)
	10		0.01	0.09 (0.00)	0.00	0.08 (0.00)	0.00	0.08 (0.00)	0.00	0.07 (0.02)
IMDb	∞	0.86 (0.00)	95.00	0.72 (0.00)	93.49	0.72 (0.00)	89.05	0.72 (0.00)	N/A	N/A
	$2,500$		1.74	0.49 (0.04)	0.22	0.42 (0.04)	7.08	0.64 (0.02)	1.69	0.63 (0.01)
	$1,000$		0.18	0.49 (0.06)	0.16	0.40 (0.05)	0.25	0.47 (0.04)	1.04	0.60 (0.02)
	750		0.07	0.43 (0.08)	0.15	0.47 (0.03)	0.15	0.47 (0.05)	0.76	0.58 (0.02)
	500		0.04	0.44 (0.02)	0.12	0.43 (0.03)	0.05	0.45 (0.05)	0.52	0.53 (0.04)
	250		0.03	0.46 (0.02)	0.11	0.43 (0.02)	0.09	0.46 (0.02)	0.32	0.55 (0.03)
	100		0.02	0.46 (0.03)	0.08	0.45 (0.03)	0.06	0.43 (0.06)	0.00	0.38 (0.03)
	50		0.01	0.43 (0.01)	0.08	0.46 (0.08)	0.04	0.45 (0.05)	0.00	0.40 (0.06)
	10		0.01	0.44 (0.07)	0.05	0.46 (0.04)	0.03	0.45 (0.07)	0.00	0.41 (0.06)
Drugs.com	∞	0.78 (0.02)	92.41	0.74 (0.01)	93.46	0.77 (0.01)	88.47	0.76 (0.01)	N/A	N/A
	$2,500$		1.62	0.37 (0.00)	0.15	0.37 (0.00)	5.59	0.62 (0.02)	0.99	0.38 (0.00)
	$1,000$		0.12	0.37 (0.00)	0.08	0.37 (0.00)	0.15	0.37 (0.00)	0.46	0.39 (0.02)
	750		0.05	0.37 (0.00)	0.07	0.37 (0.00)	0.08	0.37 (0.00)	0.38	0.37 (0.00)
	500		0.03	0.37 (0.00)	0.06	0.37 (0.00)	0.05	0.37 (0.00)	0.28	0.37 (0.00)
	250		0.02	0.37 (0.00)	0.06	0.37 (0.00)	0.05	0.37 (0.00)	0.20	0.37 (0.00)
	100		0.01	0.37 (0.00)	0.05	0.37 (0.00)	0.04	0.37 (0.00)	0.00	0.37 (0.00)
	50		0.01	0.37 (0.00)	0.04	0.37 (0.00)	0.03	0.37 (0.00)	0.00	0.37 (0.00)
	10		0.01	0.37 (0.00)	0.04	0.37 (0.00)	0.03	0.37 (0.00)	0.00	0.37 (0.00)
Amazon	∞	0.91 (0.00)	26.96	0.90 (0.00)	96.52	0.90 (0.00)	57.16	0.91 (0.00)	N/A	N/A
	$2,500$		0.57	0.70 (0.01)	0.24	0.81 (0.04)	3.44	0.87 (0.01)	0.87	0.87 (0.00)
	$1,000$		0.09	0.51 (0.01)	0.22	0.40 (0.12)	0.17	0.82 (0.01)	0.66	0.85 (0.00)
	750		0.06	0.46 (0.15)	0.20	0.38 (0.09)	0.13	0.83 (0.01)	0.46	0.84 (0.01)
	500		0.05	0.27 (0.05)	0.17	0.33 (0.00)	0.12	0.79 (0.04)	0.33	0.83 (0.00)
	250		0.04	0.32 (0.02)	0.13	0.33 (0.00)	0.14	0.33 (0.00)	0.15	0.82 (0.01)
	100		0.04	0.37 (0.08)	0.11	0.33 (0.00)	0.12	0.33 (0.01)	0.00	0.33 (0.00)
	50		0.04	0.32 (0.02)	0.10	0.33 (0.00)	0.10	0.33 (0.00)	0.00	0.33 (0.00)
	10		0.04	0.43 (0.16)	0.09	0.38 (0.09)	0.09	0.33 (0.00)	0.00	0.33 (0.00)

Table 5: BLEU scores for the training partition of each dataset and downstream macro-averaged test F_1 performance, with each of the four models using the Analytic Gaussian mechanism and the original test F_1 results provided for comparison. Test F_1 scores shown as "mean (standard deviation)", averaging over results using three random seeds. 'N/A' refers to configurations that we did not run for DP-BART-PR+, since there are no additional noisy training steps at $\varepsilon = \infty$. Higher BLEU corresponds to better performance of the rewriting model for intrinsic evaluation, higher test F_1 corresponds to better downstream performance using the rewritten dataset for training. Lower ε corresponds to better privacy.

ACL 2023 Responsible NLP Checklist

A For every submission:

☐ A1. Did you describe the limitations of your work?
Left blank.

☐ A2. Did you discuss any potential risks of your work?
Left blank.

☐ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☐ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☐ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☐ Did you run computational experiments?

Left blank.

☐ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☐ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☐ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☐ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Left blank.