SPLIT-AND-DENOISE: PROTECT LARGE LANGUAGE MODEL INFERENCE WITH LOCAL DIFFERENTIAL PRIVACY

Anonymous authors
Paper under double-blind review

ABSTRACT

Large Language Models (LLMs) shows powerful capability in natural language understanding by capturing hidden semantics in vector space. This process enriches the value of the text embeddings for various downstream tasks, thereby fostering the Embedding-as-a-Service (EaaS) business model. However, the direct transmission of text to servers poses a largely unaddressed risk of privacy leakage. To mitigate this issue, we introduce Split-N-Denoise (SnD), an innovative framework that split the model to execute the token embedding layer on the client side at minimal computational cost. This allows the client to introduce noise prior to transmitting the embeddings to the server and subsequently receive and denoise the perturbed output embeddings for downstream tasks. Our approach is designed for the inference stage of LLMs and requires no modifications to the model parameters, while also being computationally efficient on the client side. Extensive experiments demonstrate SnD's effectiveness in optimizing the privacy-utility tradeoff across various LLM architectures and diverse downstream tasks. The results reveal an significant accuracy improvement under the same privacy budget compared to the baseline, offering clients a privacy-preserving solution for local privacy protection.

1 INTRODUCTION

Large Language Models (LLMs) has shown powerful capability in natural language understanding by capturing hidden semantics in vector space. Consequently, As a result, users can leverage LLMs to obtain embeddings and subsequently apply them to their own downstream tasks, known as "embedding as a service" (EaaS). However, EaaS is typically provided as an online service, giving rise to significant privacy concerns. In particular, users may input sensitive information, such as names, phones, and email addresses, that needs to be kept hidden from the service provider. With the growing concern around the potential leakage of confidential data, certain companies, such as Samsung, have temporally prohibited the usage of online LLM services.

Recent research on privacy-preserving LLM inference investigates around two directions, cryptographic Liu & Liu (2023); Chen et al. (2022) and perturbation Du et al. (2023). Cryptographic typically employs homomorphic encryption (HE) to compute the inference result of the users' encrypted input. Unfortunately, the application of cryptographic technique is constrained by the significant computation overhead of cryptographic operations, especially on large transformer models. Perturbation provides differential privacy (DP) guarantee by adding calibrated noise to the original data. A key challenge of this approach is how to balance the utility and privacy tradeoff in a local differential privacy (LDP) setting, where users' inputs are privatized before being released to the server. Furthermore, privatization on text data is particularly difficult when the randomized algorithm is required to map text input to text output.

Split learning Gupta & Raskar (2018); Vepakomma et al. (2018) has emerged as a solution to privacy-preserving computation among two parties. During inference, the user performs affordable computation locally to obtain intermediate results (IRs), and forwards them to the service provider for subsequent operations. DP is employed to mitigate privacy leakage by injecting noises into the

IRs before sharing with the server. In the split inference setting, a crucial problem is to design an algorithm that minimizes the impact on model performance while ensuring LDP.

A notable approach involves the application of denoising techniques to conduct error correction and enhance model utility. Existing studies incorporate denoising layers on the server side, leveraging the post-processing properties of DP Nasr et al. (2020); Wang et al. (2019); Xu et al. (2022). However, the effectiveness of denoising is hindered by the fact that the server is ignorant of the injected noise levels. Driven by the limitation, a question arises: *can we improve the utility by conducting denoising on the user side, leveraging the knowledge of noise levels and raw IRs?* It's a highly non-trivial task to unconver the closed-form mapping between denoised embedding and noises as well as raw IRs, since the inputs have undergone a series of complex transformations.

In this paper, we answer this question affirmatively by proposing Split-N-Denoise (SnD), a framework that integrates split inference and denoising techniques to enhance utility under LDP bound. To minimize the computational overhead on users, we deploy only the token representation layer on client sides. A denoise model that enhances noisy embeddings using raw inputs and noise levels is pretrained on the server side and subsequently shared with the user. Once receiving the output from server, users input their private data into the denoise model to improve the utility of embeddings.

Our main contributions involve the following:

- To the best of our knowledge, this paper represents the pioneering effort in protect user's privacy during LLM inference with strong privacy guarantee. Existing research focuses on the privacy-preserving pre-training and fine-tuning for LLM, while few studies pay attention to the privacy concerns at inference stage, especially on privatizing user's input to guarantee DP. Our research introduces a framework to guarantee user's privacy with LDP while maintaining acceptable utility of the output.
- We propose SnD, a framework that integrates split inference and denoising techniques to protect user's privacy with LDP. We conduct empirical analysis to demonstrate the utility of embeddings on various downstream tasks.
- We design a novel denoising method deployed on user side. In this approach, a denoise model is pre-trained on server side using public dataset and synthetic noises. Subsequently, this trained model is deployed on the user side, where it leverages the specific noise levels and raw IRs provided by the user to enhance the embeddings.

2 Prior Works

Local Privacy for Frozen LLMs With the advent of LLMs, privacy leakage has emerged as a crucial concern. Existing literature predominantly focuses on privacy conservation throughout the entire training process, encompassing pre-training Hoory et al. (2021), fine-tuning Huang et al. (2020); Kerrigan et al. (2020); Yu et al. (2021); Lukas et al. (2023), and prompt-tuning phases Duan et al. (2023); Li et al. (2023). Yet, there is a notable dearth of research addressing local privacy during the inference phase with a fully frozen LLM. This scenario, which prohibits alterations to the model's structure and parameters, is particularly complex. Nonetheless, it holds significance in black-box API access contexts, especially for proprietary models like GPT-4. One intuitive approach involves anonymizing sensitive terms prior to LLM input and subsequently restoring them post-output Kan et al. (2023); Chen et al. (2023). However, this method, while effective for obfuscating specific entities, falls short in concealing other linguistic elements, including verbs and non-named entities. Such a limitation compromises full privacy and is unsuitable for tasks necessitating exact semantic interpretation of the altered entities, such as knowledge retrieval and text continuation Chen et al. (2023). An alternative strategy might entail input text perturbation via text-to-text privatization or synthetic data generation, preserving high-dimensional features while altering human-perceivable sequences Li et al. (2023). Specifically, the text-to-text privatization projects text into a high-dimensional vector space with a pre-determined word embedding model,adding carefully calibrated noise to the vector representation, and then reconvert it to obtain the perturbed text Feyisetan et al. (2019); Qu et al. (2021a). Yet, the mere application of this technique during inference does not guarantee a satisfactory balance between privacy and utility. Another direction employs homomorphic encyption (HE) to conduct private transformer inference Liu & Liu (2023); Chen et al. (2022), but the significant overhead renders it impractical for implementation in LLM.

Privacy-Preserving Split Learning Split learning is a novel privacy-preserving approach in distributed learning, where each client trains a segment of a deep network up to a designated "cut layer." The outputs at this layer are then forwarded to the server side, which completes the training without accessing the client's raw data. This approach facilitates forward and backward propagation without sharing raw data, ensuring the client-side local privacy Gupta & Raskar (2018); Vepakomma et al. (2018). Vepakomma et al shows that split learning surpasses federated learning and large batch synchronous SGD in achieving superior accuracy with significantly reduced client-side computational demands Gupta & Raskar (2018). Singh et al further validate its efficacy across broader experimental contexts, demonstrating that an increase in the number of clients or model dimensions gives split learning an edge over federated learning Singh et al. (2019). The advantage in its computational efficiency renders it suitable for LLM local privacy setting, where the client side executes minimal computational tasks, such as noising and denoising operations at specific segmented layers, to ensure privacy at reduced computational expenses. Meanwhile, the server handles the bulk of the model's layers. Our research serves as an initial endeavor to integrate split learning with LLM privacy concerns.

Denoising for Differential Privacy (DP) While elevated noise levels offer robust privacy protections, privacy-preserving methods inevitably compromise the model's quality Wang et al. (2019). A notable approach involves the application of denoising techniques specifically tailored for Differential Privacy (DP), incorporating a post-processing layer to enhance DP utility. Pioneering research in statistical estimation underscores the efficacy of post-processing denoising in achieving accurate private network degree distribution estimates Hay et al. (2009), and in reducing linear regression estimation errors when the ground truth is sparse Nikolov et al. (2013). Balle et al. demonstrated that denoising significantly enhances the Gaussian mechanism's accuracy in high-dimensional settings for DP algorithms with output perturbations Balle & Wang (2018). More recently, denoising mechanisms have been extended to the training of Machine Learning (ML) models, particularly Deep Neural Networks (DNNs), by applying denoising techniques to Gaussian noise-injected gradients, thereby improving the utility of privately trained ML models Wang et al. (2019). Nasr, Shokri, and Houmansadr further explored the use of scaling as a denoising strategy to optimize DP utility in Differential Privacy Stochastic Gradient Descent (DP-SGD), scaling the noisy gradients based on their usefulness Nasr et al. (2020). Subsequently, Xu et al. employed scaling and masking as post-processing denoising techniques on top of Gaussian noise-injected intermediate results in split learning, aiming to reduce the noisy neural network output's estimation error without compromising privacy Xu et al. (2022).

3 METHODOLOGY

3.1 PRELIMINARIES

3.1.1 LDP

Differential privacy (DP) Dwork (2006); Dwork et al. (2014) is considered the gold standard for data privacy. Its definition is as follows:

Definition 1 ((ϵ, δ)-Differential Privacy) *A randomized mechanism M with domain D and range R preserves (ϵ, δ)-differential privacy if and only if for any two neighboring datasets $D, D' \in D$ and for any subset $S \subseteq R$, the following inequality holds:*

$$\Pr[M(D) \in S] \leq e^{\epsilon} \Pr[M(D') \in S] + \delta$$

where ϵ is the privacy budget and δ is the failure probability.

Local differential privacy (LDP) is a particular case of DP, where the server is not trusted and data privatization is conducted by the client. For any inputs x, $x' \in D$, LDP requires a randomized mechanism M to satisfy:

$$\Pr[M(x) \in S] \leq e^{\epsilon} \Pr[M(x') \in S] + \delta \tag{1}$$

for any measurable subset subset $S \subseteq Range(M)$.

3.1.2 d_χ-PRIVACY

In the context of local privacy preservation, we employ d_χ-privacy Chatzikokolakis et al. (2013), a specialized variant of local differential privacy tailored for textual data Feyisetan et al. (2019); Qu et al. (2021a). d_χ-privacy allows to impose high probability of observing the same output for inputs with similar semantics. We state the formal definition in the following:

Definition 2 (d_χ-privacy) *For an input domain X and an output domain Y, d_χ serves as a metric space over X. A stochastic mechanism $M : X \rightarrow Y$ is said to adhere to ηd_χ-privacy if, for any two elements $x, x' \in X$, the output distributions $M(x)$ and $M(x')$ satisfy the following inequality:*

$$\frac{P(M(x) = y)}{P(M(x') = y)} \leq e^{\eta d_\chi(x,x')}, \quad \forall y \in Y,$$

where $\eta \geq 0$ is a tunable privacy parameter that modulates the level of privacy protection.

The privacy guarantee indicates that the log-likelihood ratio of producing the same outcome y is bounded by $\eta d_\chi(x, x')$ for any two possible inputs x, x'.

3.2 ARCHITECTURE

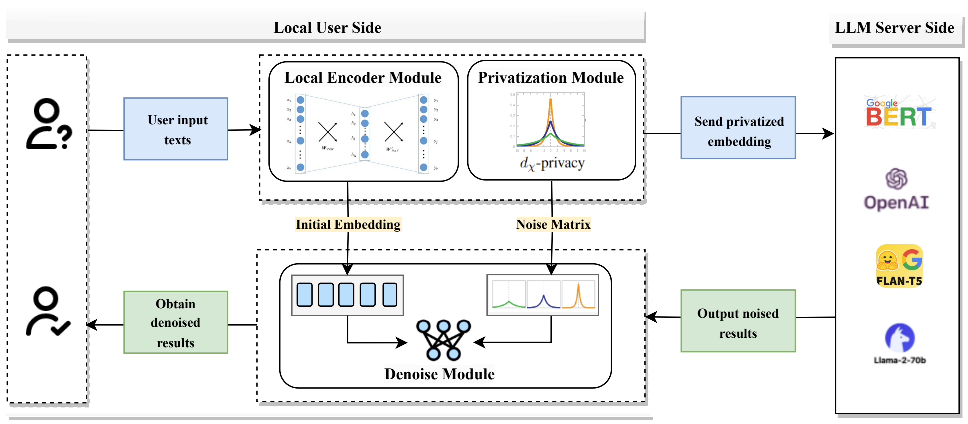


Figure 1: Overview of our privacy-preserving SnD framework.Users first obtain an initial embedding from a local encoder, followed by a noise addition via the privatization module. This privatized embedding is then transmitted to the server for processing. Upon completion, users receive a noised output, which is subsequently refined using a pre-trained denoising model to achieve an optimal balance between privacy and utility.

Denote $G : \mathcal{V}^n \rightarrow \mathbb{R}^d$ as the language model that maps n-token to embedding. In Split-N-Denoise (SnD), we split the language model G into a local encoder $G_l : \mathcal{V}^n \rightarrow \mathbb{R}^{n\times d}$ at user side and a cloud encoder $G_c : \mathbb{R}^{n\times d} \rightarrow \mathbb{R}^d$ at server side. The local encoder consists of only the token representation layer to minimize the computation cost for user, and the server performs subsequent operations on the IRs uploaded by the clients. The architecture of SnD is depicted in figure, containing four main components:

- *Local encoder module*: the user retrieves the token embeddings of their input locally.
- *Privatization module*: the token representations are privatized by the user before being transmitted to the server to satisfy LDP.
- *Cloud encoder module*: the server performs transformation on the privatized token representations and returns the embedding to user.
- *Denoise module*: user conducts local denoising on the received embedding leveraging their raw inputs and specific noise levels.

3.3 NOISE MECHANISM

We adopt d_χ-privacy to privatize the token representation layers on user side. Given an input sequence $x = [x_1, \ldots, x_n]$, the token representation layer transforms x into a vector sequence $X = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n] \in \mathbb{R}^{n \times d}$ via embedding model $E \in \mathbb{R}^{|\mathcal{V}| \times d}$, where $|\mathcal{V}|$ denotes the vocabulary size and d represents the dimensionality of the embeddings.

Assuming L_2 norm as the distance metric, the application of d_X privacy, parameterized by η, to a given word embedding $\boldsymbol{x}_t \in \mathbb{R}^d$ is realized by the addition of Laplacian noise $z \sim c \exp(-\eta||z||)$, where c is a real-valued constant Wu et al. (2017). To sample z from the Laplacian distribution, consider $z = l\boldsymbol{v}$, where l is sampled from a Gamma distribution $\Gamma(d, 1/\eta)$ and $\boldsymbol{v}$ is uniformly sampled from the unit ball B^d. Consequently, the privatized representation $M(\boldsymbol{x}_t)$ can be succinctly expressed as:

$$M(\boldsymbol{x}_t) = \boldsymbol{x}_t + \boldsymbol{z}.$$

The following theorem states that the noise mechanism $M : \mathbb{R}^d \to \mathbb{R}^d$ adheres to $\eta d_\chi-$privacy.

Theorem 1 *For any $d \geq 1$ and any $\eta > 0$, the mechanism $M : \mathbb{R}^d \to \mathbb{R}^d$ achieves $\eta d_\chi-$privacy with respect to $d_\chi(x, x') = \|x - x'\|$.*

Refer to appendix A.1 for the proof.

3.4 DENOISE MODEL

On receiving the noisy embeddings, users conduct error correction on the embeddings using their specific noises and raw inputs. Given the black-box nature of neural network transformation on the privatized token representations, we propose to train a transformer-based model for embedding denoise.

Let $\tilde{X} = [\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n]$, $Z = [\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n] \in \mathbb{R}^{n \times d}$ denote, respectively, the privatized token representations and noise matrix. After a series of operations, the server returns a noisy embedding $\boldsymbol{e}_n$ capturing the context of input token to the user. The denoise model is parameterized by a L-layer transformer decoder, $D : \mathbb{R}^{(2n+1) \times d} \to \mathbb{R}^d$:

$$\boldsymbol{e}_d = D(\boldsymbol{e}_n, \tilde{X}, Z) \tag{2}$$

The input to the denoise model H_0 is a concatenation of vectors:

$$H_0 = [\boldsymbol{e}_n; \tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n; \boldsymbol{z}_1, \ldots, \boldsymbol{z}_n] \tag{3}$$

Let $\boldsymbol{h}_t^l$ represents the hidden state for the t^{th} vector at layer l. This state is computed using the following recursive relation:

$$\boldsymbol{h}_t^l = \boldsymbol{h}_t^{l-1} + \boldsymbol{a}_t^{l-1} + \boldsymbol{m}_t^{l-1} \tag{4}$$

where

$$\boldsymbol{a}_t^{l-1} = attn^l(\boldsymbol{h}_1^{l-1}, \boldsymbol{h}_2^{l-1}, ..., \boldsymbol{h}_{2n+1}^{l-1}),\ \boldsymbol{m}_t^{l-1} = W_{proj}^l \sigma(W_{fc}^l \gamma(\boldsymbol{a}_t^l + \boldsymbol{h}_t^{l-1})) \tag{5}$$

The denoised embedding is obtained directly from the hidden state representation for $\boldsymbol{e}_n$ at the final layer:

$$\boldsymbol{e}_d = \boldsymbol{h}_0^L \tag{6}$$

We visualize the architecture of the denoise model in figure 2. Intuitively, the noisy embedding undergoes L steps to transform into the denoised embedding. In each step, the transformation is conditioned on the feature representations of raw IRs as well as specific noises.

To train a denoise model, the server samples a set of noises added to the token representations of public corpus. Subsequently, the clean embedding $\boldsymbol{e}_c$ and noisy embedding $\boldsymbol{e}_n$ are computed from, respectively, the raw and privatized token representations:

$$\boldsymbol{e}_c = G(X),\ \boldsymbol{e}_n = G(\tilde{X}) \tag{7}$$

The denoise model is trained on the above datasets with the objective to minimize the deviation between denoised and clean embeddings:

$$\min_D \mathbb{E}[\|D(\boldsymbol{e}_n, \tilde{X}, Z) - \boldsymbol{e}_c\|^2] \tag{8}$$

The pretrained model is shared with users to conduct denoising on the received embeddings locally.

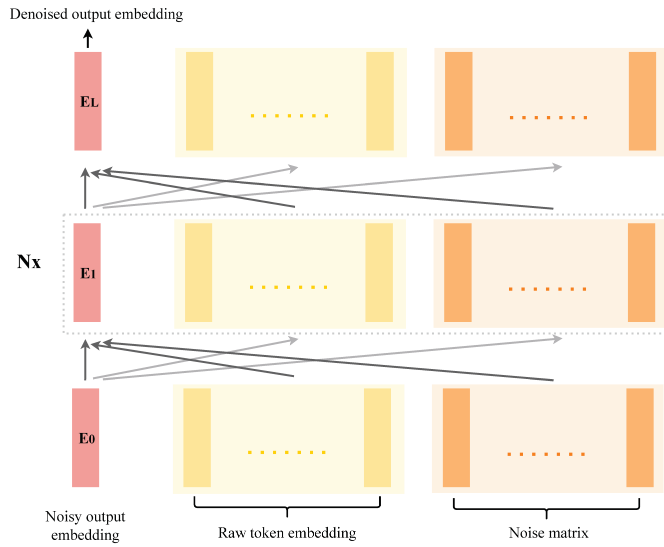


Figure 2: Architecture of denoise model.

3.5 Complexity Analysis

In this section, we analyze the communication complexity and user computation complexity of our framework.

Communication complexity: the communication cost can be broken as: (1) user uploads the token representations to the server ($O(nd)$ messages); (2) server share the embeddings with user ($O(d)$ messages). Hence, the total communication overhead is $O(nd)$.

User computation complexity: user's computation cost can be broken as: (1) retrieving token embeddings from input text ($O(n)$ complexity); (2) performing local denoising with the transformer-based model ($O(n^2dL)$ complexity Vaswani et al. (2017)). Therefore, the user's computation cost adds up to $O(n^2dL)$.

4 Experimental Results

4.1 Experiment Settup

We evaluate our framework on three classes of LLMs: Bert Devlin et al. (2018), GPT2 Radford et al. (2019), and T5 Raffel et al. (2020). The hyperparameters and model sizes of our denoise model are described in appendix A.4. We benchmark our experiments against the token embedding privatization (TokEmbPriv) method Qu et al. (2021b), where the token embeddings are perturbed by the user before sending them to the server.

To assess the performance of our approach, we employ two distinct evaluation metrics: (1) similarity with $\boldsymbol{e}_c$: we compute the mean square error (MSE) and cosine similarity (COS) between $\boldsymbol{e}_c$ and $\boldsymbol{e}_d$, the clean and privatized embeddings, to quantify the extent of data variations induced by the perturbation process; (2) performance on downstream tasks: we utilize accuracy scores (ACC) and area under the roc curve (AUC) to gauge the utility of the embeddings on downstream tasks.

4.2 Datasets

To train the denoise model, we use the combination of 20 datasets to better mimic the generalized training scenarios, including TweetEval Offensive Barbieri et al. (2020), Hate Speech 18 de Gibert et al. (2018), Health Fact Kotonya & Toni (2020),Daily Dialogue Li et al. (2017), etc. See the full list of datasets we used in Appendix A.2.

We test our denoising performance on a collection of downstream tasks:

- Sentence classification: CoLA Warstadt et al. (2019)
- Pair similarity: Quora Question Pairs (QQP) Chen et al. (2018), MSR Paraphrase Corpus (MRPC) Dolan & Brockett (2005)
- Recognizing Textual Entailment (RTE) Dagan et al. (2006); Bar-Haim et al. (2006); Giampiccolo et al. (2007); Bentivogli et al. (2009)

Refer to appendix A.3 for the evaluation details.

4.3 Experiment Results

4.3.1 Performance on downstream task

We record the performance on various downstream task in terms of accuracy (ACC) under varing η in table 4.3.1, 4.3.1 and 4.3.1. The utility is benchmarked against the case without any noise injection, denoted by $\eta = \infty$. One important observation is that our framework maintains acceptable accuracy compared with the non-privatized setting. Noted that we perform evaluation on the embeddings from pre-trained model without any fine-tuning, and thus there's a gap between the accuracy in our results and the SOTA benchmarks.

	DistillBert (66m)			Bert Base (110m)			Bert Large (340m)		
eta	100	500	∞	100	500	∞	100	500	∞
CoLA	0.688	0.689	0.701	0.670	0.670	0.751	0.691	0.683	0.744
QQP	0.618	0.626	0.683	0.618	0.656	0.728	0.637	0.631	0.706
MRPC	0.670	0.672	0.674	0.672	0.670	0.664	0.684	0.689	0.659
RTE	0.578	0.580	0.592	0.563	0.566	0.569	0.585	0.549	0.599

Table 1: Accuracies on downstream tasks for BERT

4.3.2 Comparison with baseline

In Figure 4.3.2, we assess and compare the performance of three model families using three distinct metrics: AUC (Area Under Curve) for overall efficacy, MSE (Mean Squared Error) for model loss, and Cosine Similarity between the initial and recovered embeddings to gauge the effectiveness of our denoising mechanism. Against each metric, we compare our techniques with the baseline method TokEmbPriv Qu et al. (2021b).

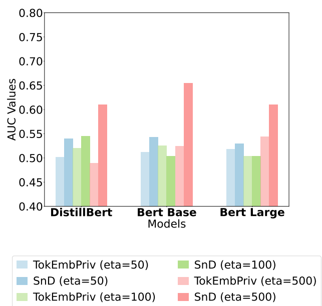
For the three model families, we selected three distinct eta levels for experimentation, given the varying noise tolerance of each model. Specifically, for the Bert models, we set eta to 50, 100, and 500; for the GPT models, the values were 1, 100, and 1000; and for the T5 models, we chose 0.1, 1, and 10. For each model family, a representative task was selected. Our observations indicate that our method outperforms the baseline in terms of AUC in roughly 77% of the tested combinations, highlighting our model's ability to retain utility. Moreover, in approximately 90% of the combinations, we noted substantially reduced MSE values and enhanced Cosine Similarity scores. This suggests our technique's proficiency in restoring the original attributes of the noised embedding after procuring the perturbed results from the server.

	T5 Small (60m)				T5 Base (220m)				T5 Large (770m)			
eta	0.001	0.01	1	∞	0.001	0.01	1	∞	0.001	0.01	1	∞
CoLA	0.69	0.69	0.69	0.71	0.70	0.69	0.69	0.73	0.69	0.69	0.69	0.74
QQP	0.68	0.69	0.68	0.71	0.63	0.69	0.69	0.72	0.63	0.62	0.70	0.71
MRPC	0.68	0.68	0.68	0.69	0.68	0.68	0.68	0.71	0.68	0.68	0.68	0.71
RTE	0.51	0.52	0.54	0.58	0.52	0.54	0.55	0.56	0.55	0.56	0.53	0.58

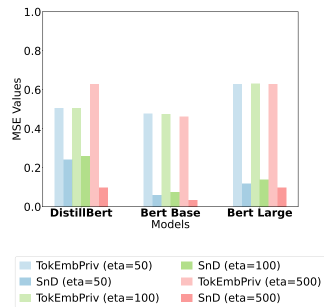
Table 2: Accuracies on downstream tasks for T5

	GPT2 Small (120m)			GPT2 Medium (345m)			GPT2 large (774m)			GPT2 Xlarge (1.5b)	
eta	1	100	∞	1	100	∞	1	100	∞	100	∞
CoLA	0.688	0.684	0.709	0.688	0.692	0.728	0.691	0.69	0.724	0.690	0.766
QQP	0.617	0.600	0.716	0.626	0.632	0.711	0.615	0.631	0.721	0.592	0.721
MRPC	0.688	0.681	0.720	0.684	0.686	0.710	0.674	0.684	0.701	0.676	0.705
RTE	0.535	0.549	0.579	0.556	0.567	0.583	0.567	0.596	0.582	0.574	0.592

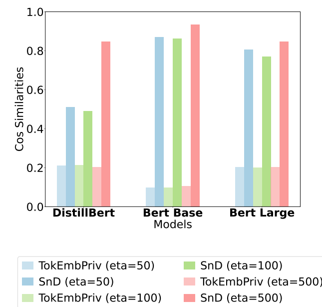
Table 3: Accuracies on downstream tasks for GPT2



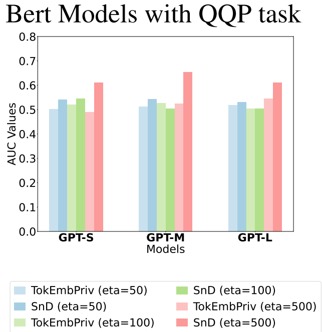
*(a) AUC Comparison for Bert Models with QQP task



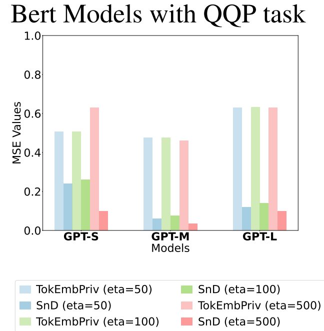
*(b) MSE Comparison for Bert Models with QQP task



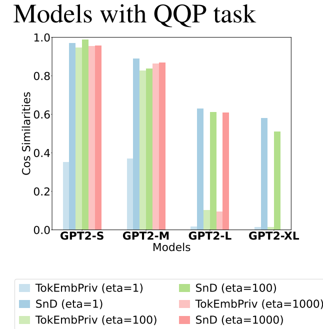
*(c) Cos Similarities for Bert Models with QQP task



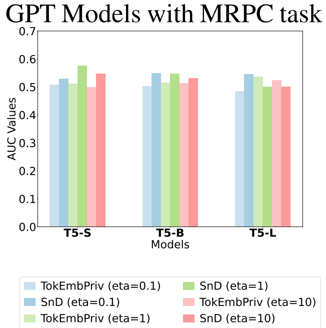
*(d) AUC Comparison for GPT Models with MRPC task



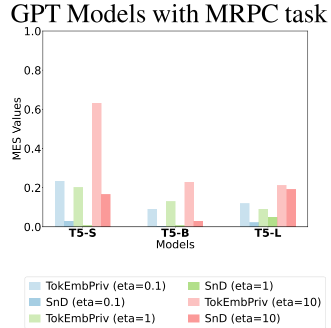
*(e) MSE Comparison for GPT Models with MRPC task



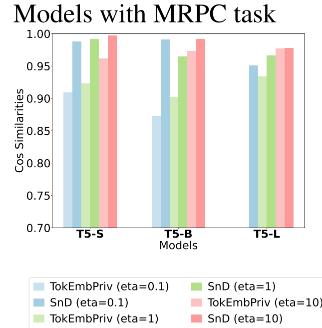
*(f) Cos Similarities for GPT Models with MRPC task



*(g) AUC Comparison for T5 Models with MRPC task



*(h) MSE Comparison for T5 Models with MRPC task



*(i) Cos Similarities for T5 Models with MRPC task

Figure 3: Baseline performance comparison of 3 model families under multiple eta levels

5 Discussion and Future Work

Scalability to larger language model: our experiments primarily focused on language models ranging from 100MB to 1GB in size. We also tested our approach on larger language model, such as LLaMa and OPT-6.7B. While we observed substantial improvements in terms of MSE and COS of the embeddings compared to the baseline, we discovered that the accuracy on downstream tasks still requires further enhancement. We suspect that the inputs undergo significantly more intricate transformations in these larger language models, necessitating the use of more sophisticated noise and denoising mechanisms.

Reduce user computation cost: local denoising constitutes a major component of user's computation overhead. We observe that the size of denoise model, and thus the user computation cost, scale with the underlying LLM. For those users with limited computation resource, it's crucial to design a lightweight denoise mechanism with minimal computation cost.

Sequence-to-sequence (S2S) inference: it's of great interest to extend our EaaS framework to S2S inference model. One important obstacle of the generalization is the noise amplification issue with S2S model. In particular, S2S relies on the auto-regressive mechanism, where the prediction of previous token is taken as an input to the next token. Therefore, the error from the previous prediction would exaggerate the deviation of the following tokens. A universal denoise model might be insufficient to correct the errors in the generated sequence.

6 Conclusion

This paper proposes SnD, a framework that employs split inference and denoising techniques to protect LLM inference with LDP. We split the language model to deploy the token representation layer on user side. User perturbs the token embeddings to guarantee d_χ-privacy before transmitting them to the server. To improve the utility of embeddings, user conducts local denoising with a pre-trained model leveraging the raw token representations and specific noises. The empirical studies show that: (1) SnD performs better in maintaining the utility of embeddings compared with baseline method. (2) The denoise model significantly reduce the noises in the output, as demonstrated by improvements in terms of both cosine similarity (COS) and mean squared error (MSE). Our study opens up new possibilities for privacy-preserving LLM inference, in terms of scalabibility to larger LLM, optimizating user computation cost, and extension to sequence-to-sequence inference model.

References

Tiago A. Almeida, Jose Maria Gomez Hidalgo, and Akebo Yamakami. Contributions to the Study of SMS Spam Filtering: New Collection and Results. In *Proceedings of the 2011 ACM Symposium on Document Engineering (DOCENG'11)*, 2011.

Borja Balle and Yu-Xiang Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising, 2018.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop*, 2006.

Francesco Barbieri, Jose Camacho-Collados, Luis Espinosa-Anke, and Leonardo Neves. TweetEval:Unified Benchmark and Comparative Evaluation for Tweet Classification. In *Proceedings of Findings of EMNLP*, 2020.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth pascal recognizing textual entailment challenge. In *Proceedings of the TAC 2009 Workshop*, 2009.

Iñigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. Efficient Intent Detection with Dual Sentence Encoders. In *Proceedings of the 2nd Workshop on NLP for ConvAI - ACL 2020*, 2020. URL `https://arxiv.org/abs/2003.04807`. Data available at https://github.com/PolyAI-LDN/task-specific-datasets.

Kostas Chatzikokolakis, Miguel Andrés, Nicolás Bordenabe, and Catuscia Palamidessi. Broadening the scope of differential privacy using metrics. 07 2013. ISBN 978-3-642-39076-0. doi: 10.1007/978-3-642-39077-7_5.

Tianyu Chen, Hangbo Bao, Shaohan Huang, Li Dong, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. The-x: Privacy-preserving transformer inference with homomorphic encryption. *arXiv preprint arXiv:2206.00216*, 2022.

Yu Chen, Tingxin Li, Huiming Liu, and Yang Yu. Hide and seek (has): A lightweight framework for prompt privacy protection, 2023.

Zihan Chen, Hongbo Zhang, Xiaoji Zhang, and Leqi Zhao. Quora question pairs, 2018. URL `https://www.kaggle.com/c/quora-question-pairs`.

Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges*, 2006.

Ona de Gibert, Naiara Perez, Aitor García-Pablos, and Montse Cuadros. Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pp. 11–20, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5102. URL `https://www.aclweb.org/anthology/W18-5102`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*, 2005.

Minxin Du, Xiang Yue, Sherman SM Chow, Tianhao Wang, Chenyu Huang, and Huan Sun. Dp-forward: Fine-tuning and inference on language models with differential privacy in forward pass. *arXiv preprint arXiv:2309.06746*, 2023.

Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models, 2023.

Cynthia Dwork. Differential privacy. In *International colloquium on automata, languages, and programming*, pp. 1–12. Springer, 2006.

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

Natasha Fernandes, Mark Dras, and Annabelle McIver. Generalised differential privacy for text document processing. In *Principles of Security and Trust: 8th International Conference, POST 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6–11, 2019, Proceedings 8*, pp. 123–148. Springer International Publishing, 2019.

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy- and utility-preserving textual analysis via calibrated multivariate perturbations, 2019.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, 2007.

Aaron Gokaslan*, Vanya Cohen*, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019.

Giovanni Grano, Andrea Di Sorbo, Francesco Mercaldo, Corrado A Visaggio, Gerardo Canfora, and Sebastiano Panichella. Software applications user reviews. 2017.

Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents, 2018.

Harsha Gurulingappa, Abdul Mateen Rajput, Angus Roberts, Juliane Fluck, Martin Hofmann-Apitius, and Luca Toldo. Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of Biomedical Informatics*, 45:885–892, 2012. doi: https://doi.org/10.1016/j.jbi.2012.04.008. URL `http://www.sciencedirect.com/science/article/pii/S1532046412000615`.

Michael Hay, Chao Li, Gerome Miklau, and David Jensen. Accurate estimation of the degree distribution of private networks. In *2009 Ninth IEEE International Conference on Data Mining*, pp. 169–178. IEEE, 2009.

Shlomo Hoory, Amir Feder, Avichai Tendler, Alon Cohen, Sofia Erell, Itay Laish, Hootan Nakhost, Uri Stemmer, Ayelet Benjamini, Avinatan Hassidim, and Yossi Matias. Learning and evaluating a differentially private pre-trained language model. pp. 21–29, 01 2021. doi: 10.18653/v1/2021.privatenlp-1.3.

Yangsibo Huang, Zhao Song, Danqi Chen, Kai Li, and Sanjeev Arora. Texthide: Tackling data privacy in language understanding tasks, 2020.

Zhigang Kan, Linbo Qiao, Hao Yu, Liwen Peng, Yifu Gao, and Dongsheng Li. Protecting user privacy in remote conversational systems: A privacy-preserving framework based on text sanitization, 2023.

Gavin Kerrigan, Dylan Slack, and Jens Tuyls. Differentially private language models benefit from public pre-training, 2020.

Neema Kotonya and Francesca Toni. Explainable automated fact-checking for public health claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7740–7754, Online, November 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-main.623`.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. Dailydialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of The 8th International Joint Conference on Natural Language Processing (IJCNLP 2017)*, 2017.

Yansong Li, Zhixing Tan, and Yang Liu. Privacy-preserving prompt tuning for large language model services, 2023.

Xuanqi Liu and Zhuotao Liu. Llms can understand encrypted prompt: Towards privacy-computing friendly transformers. *arXiv preprint arXiv:2305.18396*, 2023.

Nils Lukas, Ahmed Salem, Robert Sim, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Analyzing leakage of personally identifiable information in language models. *arXiv preprint arXiv:2302.00539*, 2023.

P. Malo, A. Sinha, P. Korhonen, J. Wallenius, and P. Takala. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65, 2014.

Julian McAuley and Jure Leskovec. Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pp. 165–172, 2013.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

Milad Nasr, Reza Shokri, and Amir houmansadr. Improving deep learning with differential privacy using gradient encoding and denoising, 2020.

Aleksandar Nikolov, Kunal Talwar, and Li Zhang. The geometry of differential privacy. In *Proceedings of the forty-fifth annual ACM symposium on Theory of Computing*. ACM, jun 2013. doi: 10.1145/2488608.2488652. URL `https://doi.org/10.1145%2F2488608.2488652`.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. Natural language understanding with privacy-preserving BERT. In *Proceedings of the 30th ACM International Conference on Information &amp Knowledge Management*. ACM, oct 2021a. doi: 10.1145/3459637.3482281. URL `https://doi.org/10.1145%2F3459637.3482281`.

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. Natural language understanding with privacy-preserving bert. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pp. 1488–1497, 2021b.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv e-prints*, pp. arXiv:1606.05250, 2016.

Emily Sheng and David Uthus. Investigating Societal Biases in a Poetry Composition System, 2020.

Abhishek Singh, Praneeth Vepakomma, Otkrist Gupta, and Ramesh Raskar. Detailed comparison of communication efficiency of split learning and federated learning, 2019.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013. URL `https://www.aclweb.org/anthology/D13-1170`.

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

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data, 2018.

Bao Wang, Quanquan Gu, March Boedihardjo, Farzin Barekat, and Stanley J. Osher. Dp-lssgd: A stochastic optimization method to lift the utility in privacy-preserving erm, 2019.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey F. Naughton. Bolt-on differential privacy for scalable stochastic gradient descent-based analytics, 2017.

Hang Xu, Aritra Dutta, Weiran Liu, Xin Li, and Panos Kalnis. Denoising differential privacy in split learning. 2022.

Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021.

Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. Character-level Convolutional Networks for Text Classification. In *NIPS*, 2015.

A Appendix

A.1 Proof of Theorem 1

To prove the theorem, we first demonstrate that the noise follows Laplacian distribution:

Lemma 1 *By sampling l from Gamma distribution $\Gamma(d, 1/\eta)$, and $\boldsymbol{v}$ from the unit ball B^d, the vector $z = l\boldsymbol{v}$ can be released as d-dimensional Laplacian $z \sim c\exp(-\eta\|z\|)$.*

Proof 1 *The proof follows by changing variables to spherical coordinates and showing that the Laplacian can be expressed as the product of $\boldsymbol{v}$ and l. See, for instance, Lemma 4 in Fernandes et al. (2019).*

We can now proceed to the proof of theorem 1.

Proof 2 *Plugging in the probability density function of z, it holds that:*

$$\frac{P(M(x)=y)}{P(M(x')=y)} = \frac{P(z=y-x)}{P(z=y-x')} = \exp(\eta(\|y-x'\| - \|y-x\|)) \leq \exp(\eta(\|x'-x\|)) \quad (9)$$

, for any $\eta > 0$.

A.2 Details of datasets

A.2.1 Dataset to Train Denoise Model

SQuAD: The Stanford Question Answering Dataset (SQuAD) is a reading comprehension dataset, with questions posed by crowdworkers based on a set of Wikipedia articles. The answer to every question is a segment of text from the corresponding article, or the question might be unanswerable Rajpurkar et al. (2016).

AG News: This dataset contains more than 1 million news articles, categorizing text into classes like sports, business, and tech Zhang et al. (2015).

Financial Phrasebank: Comprising sentiments in the financial domain, specifically with sentences where all annotators concur. It is primarily for 3-class sentiment analysis Malo et al. (2014).

Banking77: It contains online banking queries annotated with their corresponding intents, focusing on fine-grained single-domain intent detection Casanueva et al. (2020).

Health Fact: A comprehensive dataset for explainable automated fact-checking of public health claims Kotonya & Toni (2020).

Poem Sentiment: A dataset for 4-class sentiment analysis on poem verses from Project Gutenberg Sheng & Uthus (2020).

Tweet Eval - Sentiment: Containing tweets for sentiment analysis Barbieri et al. (2020).

Tweet Eval - Emotion: Comprising tweets labeled with specific emotions Barbieri et al. (2020).

Tweet Eval - Hate: A dataset to classify tweets containing hate speech Barbieri et al. (2020).

Tweet Eval - Offensive: A dataset for classifying tweets deemed offensive Barbieri et al. (2020).

ADE Corpus V2: A dataset for classification if a sentence discusses Adverse Drug Reaction or not. This dataset also extracts the relation between Adverse Drug Event and Drug Gurulingappa et al. (2012).

Hate Speech18: Dedicated to detecting hate speech in texts extracted from Stormfront, a white supremacist forum de Gibert et al. (2018).

SMS Spam: Comprising SMS labeled texts, this dataset is utilized to identify spam messages Almeida et al. (2011).

Daily Dialog: A high-quality multi-turn dialog dataset contains dialogues derived from daily conversations Li et al. (2017).

Yelp Review Full: Comprising reviews from Yelp for text classification. This dataset is extracted from the Yelp Dataset Challenge 2015 data Zhang et al. (2015).

App Reviews: A dataset of user reviews of Android applications belonging to different categories Grano et al. (2017).

Amazon Polarity: Contains reviews from Amazon, including product and user information, ratings, and a text review McAuley & Leskovec (2013); Zhang et al. (2015).

Rotten Tomatoes: A movie review dataset used for sentiment analysis. This dataset comprises reviews from the Rotten Tomatoes website (Pang & Lee, 2005).

Wikitext: A collection of over 100 million tokens extracted from the set of verified Good and Featured articles on Wikipedia. Used for language modeling and other NLP tasks (Merity et al., 2016).

OpenWebText: An open-source collection of web articles, modeled after the dataset used in the original "GPT" work (Gokaslan* et al., 2019).

A.2.2 Dataset for Downstream Tasks

QQP: The Quora Question Pairs2 dataset consists of question pairs to determine semantic equivalence Chen et al. (2018).

RTE: The Recognizing Textual Entailment (RTE) datasets aggregates multiple Recognizing Textual Entailment challenges, determining if texts entail each other. It combines data from several RTE challenges Dagan et al. (2006); Bar-Haim et al. (2006); Giampiccolo et al. (2007); Bentivogli et al. (2009).

SST2: The Stanford Sentiment Treebank contains movie review sentences labeled for sentiment, aiming to predict positive or negative sentiments Socher et al. (2013).

A.3 Evaluation of Downstream Tasks

We follow the steps below to conduct evaluation on downstream tasks:

- Obtain the embeddings of text in training and testing datasets via privacy-preserving LLM inference framework.
- Train a classification model on the privatized (denoised) embeddings from training set.
- Test the performance of classification model on the privatized (denoised) embeddings from testing set.

A.4 Hyperparameters of Denoise Model

The hyperparameters of denoise model are represented as followed:

- d_{model}: Dimension of input embeddings and hidden states.
- d_{ff}: Hidden dimension in the feed forward network.
- d_{kv}: Dimension of each head in the multi-head attention layer.
- n_{head}: Number of heads in the multi-head attention layer.
- L: Number of layers.

Table A.4 lists the hyperparameters for each denoise model.

	d_{model}	d_{ff}	d_{kv}	n_{head}	L
DistillBert	768	768	240	6	3
Bert Base	768	1024	240	8	6
Bert Large	1024	1024	256	8	6
T5 Small	512	512	240	6	6
T5 Base	768	768	256	8	6
T5 Large	1024	1024	256	8	6
GPT2 Small	768	768	240	8	6
GPT2 Medium	1024	1024	256	8	6
GPT2 Large	1280	1280	256	8	6
GPT2 XLarge	1600	1600	256	10	6

Table 4: Hyperparameters of denoise models