Sanitizing Sentence Embeddings (and Labels) for Local Differential Privacy

Minxin Du
Chinese Univ. of Hong Kong
dm018@ie.cuhk.edu.hk

Xiang Yue
The Ohio State University
yue.149@osu.edu

Sherman S. M. Chow*
Chinese Univ. of Hong Kong
sherman@ie.cuhk.edu.hk

Huan Sun
The Ohio State University
sun.397@osu.edu

ABSTRACT

Differentially private (DP) learning, notably DP stochastic gradient descent (DP-SGD), has limited applicability in fine-tuning gigantic pre-trained language models (LMs) for natural language processing tasks. The culprit is the perturbation of gradients (as gigantic as entire models), leading to significant efficiency and accuracy drops.

We show how to achieve metric-based *local* DP (LDP) by sanitizing (high-dimensional) sentence embedding, extracted by LMs and *much smaller* than gradients. For potential utility improvement, we impose a consistency constraint on the sanitization. We explore two approaches: One is brand new and can *directly* output consistent noisy embeddings; the other is an upgradation with *post-processing*. To further mitigate "the curse of dimensionality," we introduce two trainable linear maps for mediating dimensions without hurting privacy or utility. Our protection can effectively defend against privacy threats on embeddings. It also naturally extends to inference.

Our experiments[1] show that we reach the non-private accuracy under properly configured parameters, *e.g.*, 0.92 for SST-2 with a privacy budget $\epsilon = 10$ and the reduced dimension as 16. We also sanitize the label for LDP (with another small privacy budget) with limited accuracy losses to fully protect every sequence-label pair.

CCS CONCEPTS

• **Security and privacy → Data anonymization and sanitization**; **Privacy-preserving protocols**; **Privacy protections**.

KEYWORDS

Local Differential Privacy, Natural Language Processing, Pre-trained Language Models, Privacy-preserving NLP, Sentence Embeddings

ACM Reference Format:
Minxin Du, Xiang Yue, Sherman S. M. Chow, and Huan Sun. 2023. Sanitizing Sentence Embeddings (and Labels) for Local Differential Privacy. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3543507.3583512

1 INTRODUCTION

Recent advancement in deep learning has led to notable success in natural language processing (NLP), remarkably boosting gigantic pre-trained language models (LMs) such as BERT [15] and GPT [45] and pushing the state of the art in countless NLP tasks. But it comes with a catch. Even publicly available pre-training corpora may have private information (*e.g.*, SSNs, addresses) [10], let alone those contributed by individuals for fine-tuning sensitive tasks. For example, Carlini *et al.* [9, 10] show that LMs can (unintentionally) "memorize" pre-training data, thus vulnerable to membership inference attacks (MIAs) [47] – whether an example is used for training. Even worse, they extract verbatim text sequences with only black-box access to GPT-2. Lehman *et al.* [28] can recover patient names and related conditions from BERT fine-tuned on a private medical corpus.

Differential privacy (DP) [16] limits the impact of any individual's contribution, hence mitigating MIAs or data extraction for an adversary with any prior knowledge. It can also complement cryptographic solutions, *e.g.*, inference (single [39] or multi-server [52]) or secure multi-party computation [7, 25]. To train models with DP, a classic approach is differentially-private stochastic gradient descent (DP-SGD) [1]: For each step, it first clips per-example gradients in a batch and then adds Gaussian noise to the aggregated one. Due to its popularity and generalizability, it has been integrated into mainstream machine/deep learning frameworks, such as Opacus for PyTorch. Applying it to fine-tune LM-based NLP pipelines attains *example-level* privacy [29, 63, 64], assuming each individual contributes only one training example (or a sequence-label pair).

Unfortunately, DP-SGD often adopts a *trusted* party or utilizes secure aggregation [11, 38] with extra costs and trust assumptions [50] to curate individuals' sensitive training data, offering *central* DP [1] at its core. Also, computing and storing per-example gradients as large as entire pipelines (*e.g.*, >110M parameters for BERT-base [15]) are costly, making it challenging to strike a nice privacy-utility balance. For example, the slowdown can be up to 100× of standard training [9], and the averaged accuracy of four NLP tasks fine-tuned by DP-SGD at a moderate privacy regime is 68.5% vs. 91.8% without DP [63, Table 4]. Last but not least, due to the gradient perturbation in back-propagation, DP-SGD cannot be extended to protect the *test* data or defend against privacy attacks [41, 48] beyond MIAs.

1.1 Technical Overview and Challenges

Local DP for (high-dimensional) text data. We consider a more practical setting: Individuals can perturb their data locally to ensure *local* DP (LDP) [27] before being shared with an *untrusted* server for fine-tuning/inference, which naturally fits federated learning [34]. Yet, the standard LDP [27] may be too strong, "remembering almost nothing" about the inputs, *e.g.*, no matter how unrelated the two

[*] Corresponding author is from The Chinese University of Hong Kong, Hong Kong.
[1] Our code is available at: https://github.com/xiangyue9607/Sentence-LDP.

WWW '23, April 30 – May 4, 2023, Austin, Texas, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583512

input sequences are, the output distributions are statistically similar. We thus employ a generalized notion: metric-LDP [3], offering *heterogeneous* protection for different input pairs; the distinguishability of outputs relies on the distance between inputs under a suitable distance metric. Metric-LDP is useful when the "semantics" of inputs matter. It has been widely studied in NLP [23, 65] or others [4, 58].

Sanitizing sentence embeddings for keeping context. An active line of research [23, 44, 65] "sanitizes" text by token-wise redaction. These designs are *non-contextualized* since they process tokens independently: perturbing token embeddings (*i.e.*, real-valued high-dimensional vectors) by random noise, then mapped back to tokens via post-processing [23, 44], or directly sampling "replacements" from a discrete token universe (to avoid the dimensional curse) [65]. Yet, sanitized text, being human-readable, may still convey private information. To keep context for maintaining utility, (sensitive) tokens could be sanitized to themselves or semantically-similar ones with high probabilities [65]. Qu *et al.* [44] further study fine-tuning on noisy token embeddings, yet the task accuracy is far from that of sanitized text [23, 65] even at a low privacy regime ($\epsilon = 125$), *e.g.*, 0.56 vs. 0.83 for SST-2. In short, they only ensure metric-LDP at the *token level* [23, 44, 65] and lose the context. Directly extending them to the sequence level degrades privacy by the sequence length n.

Sentence embeddings, "aggregated" from n token embeddings at the output of LMs, have much lower dimensions, *e.g.*, 768 vs. >110M for gradients (or $n \times 768$ for token embeddings) of BERT-base [15]. Sanitizing them for *sequence-level* metric-LDP is more feasible since the noise magnitude is scaled with the dimensionality; the signal-to-noise ratio becomes much smaller when perturbing gradients locally for the same privacy level. As a side benefit, it can effectively mitigate embedding-based attacks under the same threat models as [48] (see Section 4.4), *e.g.*, inverting raw text or inferring sensitive attributes from (noisy) embeddings. It also naturally extends protection to *inference*. Efficiency-wise, it consumes less time and memory than DP-SGD computing and storing per-example gradients.

Sentence embedding normalization is beneficial for fine-tuning NLP pipelines (*e.g.*, avoiding overfitting, faster convergence [2, 66]). It implies a *consistency* constraint: The sanitized embeddings should be normalized like the raw ones. To our knowledge, we are the first to explore consistency in sanitizing sentence embeddings for fine-tuning/inference of NLP pipelines, although it is a common tool in simpler traditional statistical analytics [26, 55] under LDP. We investigate two complementary approaches: i) *directly* sampling noisy "replacements" from a normalized sphere using the Purkayastha mechanism[2] [58] or ii) *post-processing* the outputs of the generalized *planar Laplace* (PL) [60] that has been used to perturb token embeddings [23, 44] (see Section 3.2). Yet, they still suffer from the "curse of dimensionality," *e.g.*, the Purkayastha mechanism has only been shown to work well for 2- or 3-d spatial/temporal data [58], while sentence embeddings can easily be 768-d (or higher) [15].

Techniques for mitigating the curse by dimension reduction while not hurting much utility are instrumental in many fields. Random projection [60] is an efficient approach, which samples a random *fixed* linear map (from certain distributions) for reducing feature vectors' dimensions while keeping their raw geometry. We adapt it to NLP pipelines (Figure 1) by making it *trainable* for potential better utility, coupled with an extra post-processing map to "restore" the dimension to make it compatible with the raw pipelines again.

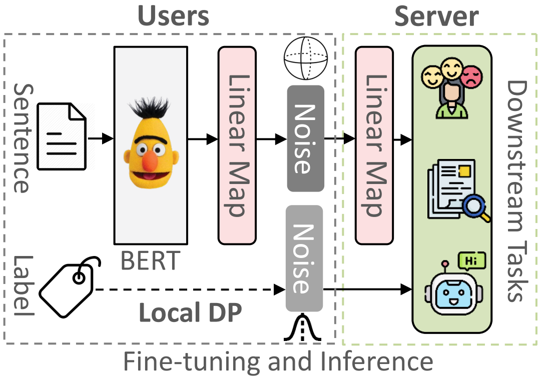

Figure 1: Sanitizing sentence embeddings (and labels) under local DP for BERT-based NLP pipelines

Sanitizing labels for full protection. Typically, each individual has a sequence-label pair for fine-tuning. To fully protect *every* pair (*cf.* DP-SGD only "hides" any single pair among the entire training data), we also let individuals sanitize their labels locally. Since labels are often discrete, we can use randomized response (RR) [56] if the label space is small; otherwise, we additionally prune the label space with prior knowledge (*e.g.*, obtained via multi-stage training [24]) before invoking RR. For better utility, such label sanitization is not mandatory if the labels are deemed non-sensitive (*e.g.*, binary classification tasks) or even absent for self-supervised learning.

1.2 Our Contributions

Motivated by the inherent shortcomings of gradient perturbation (*e.g.*, the poor generalizability to gigantic models and the vulnerability to attacks beyond MIAs), we initiate a study of sanitizing sentence embeddings for fine-tuning/testing LM-based pipelines.

i) We achieve metric-LDP [3] to enable heterogeneous protection. We impose a consistency constraint on our sanitization, borrowing the wisdom of normalizing sentence embedding for robustness [2].

ii) We propose two instantiations from the Euclidean and angular distances. The first one is brand new in NLP, which utilizes the Purkayastha mechanism (previously used for only 2-/3-dimensional data [58]). The other is upgraded from the generalized planar Laplace mechanism [60] with *post-processing*. We strategically apply two trainable maps in pipelines to mediate the dimension curse.

iii) We are the first to protect labels (for fine-tuning) with LDP, in contrast to prior arts [23, 44, 65]. We empirically show that randomized response (or its improvement [24]) generally works well.

iv) We conduct experiments on three representative NLP tasks. The results show that our LDP approaches with suitable parameters can even achieve the non-private task accuracy, outperforming DP-SGD, and effectively thwart privacy threats to embeddings [41, 48].

[2] As a metric-based notion, the sampling probability decreases exponentially with some distance metric as an exponential mechanism variant in a *token*-level metric-LDP work [65] (angular vs. Euclidean). An important difference is that ours targets a *continuous* space and needs a different normalization factor in its probability distribution.

2 PRELIMINARIES

2.1 Pre-trained LMs and Sentence Embeddings

Modern LMs, such as BERT [15] and GPT [45], are often pre-trained on enormous (public) self-labeled corpora, *e.g.*, Wikipedia. They are built atop the transformer architecture [53], enabling them to have dozen of identical layers/encoders with huge capacity. Later, they can be adapted to various NLP tasks (*e.g.*, sentiment analysis and question answering) by fine-tuning on much smaller, task-specific datasets. Such a pretrain-then-finetune paradigm avoids training a new model for each task from scratch while achieving remarkable performance gains compared to approaches without using LMs.

Let $X = \langle x_i \rangle_{i=1}^{n}$ be a sequence of n tokens or sub-words, typically obtained by splitting a sentence by the WordPiece tokenization [61]. With an embedding layer stacked before LMs, $x_{i\in[n]}$ is first mapped to a vector in $\mathbb{R}^m$, then processed inside LMs by per-layer (query, key, and value) weights in $\mathbb{R}^{m\times m}$. The hidden embedding matrix in $\mathbb{R}^{n\times m}$ is reduced to a sentence embedding $\Phi(X) \in \mathbb{R}^m$ at the LM output. The details of transformer-based LMs can be found in [53]; we just treat LMs as a black-box "oracle" to extract sentence-level features for fine-tuning/testing different downstream task layers.

This work studies BERT [15] as an example for its popularity [65]. BERT employs a masked language modeling objective to predict randomly "masked" tokens in a sequence conditioned on all the others during pre-training, allowing it to capture bidirectional contexts and outperform non-contextualized token embedding models (*e.g.*, GloVe [43]) or unidirectional GPT [45]. For BERT, common reducing methods to derive $\Phi(X)$ include mean pooling [46] (computing the average of n hidden embeddings) or just taking the last embedding corresponding to a special token `[CLS]` for classification [43].

2.2 (Local) Differential Privacy

DP [16] is a rigorous privacy guarantee, regardless of an adversary's auxiliary knowledge. It ensures that a randomized mechanism $\mathcal{M}$ behaves similarly on any two neighboring datasets $\mathcal{X} \simeq \mathcal{X}'$ differing in only one individual's contribution (*e.g.*, a sequence). Formally:

Definition 1. *Let $\epsilon \geq 0, 0 \leq \delta \leq 1$ be two privacy parameters. $\mathcal{M}$ fulfills (ϵ, δ)-DP, if $\forall \mathcal{X} \simeq \mathcal{X}'$ and any output set $O \subseteq \text{Range}(\mathcal{M})$,*

$$\Pr[\mathcal{M}(\mathcal{X}) \in O] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(\mathcal{X}') \in O] + \delta.$$

If $\delta = 0$, then we say that $\mathcal{M}$ is ϵ-DP or pure DP.

There are two popular DP settings, *central* and *local*. In central DP [16], a *trusted* curator can access all individuals' raw data, process the data by $\mathcal{M}$ with random noise for DP, and release the noisy outputs. Local DP (LDP) [27] eliminates the curator by letting individuals perturb their data locally before being shared: It offers stronger protection but makes analytics on noisy data less accurate.

Definition 2. *Let $\epsilon \geq 0$ be a privacy parameter. $\mathcal{M}$ is ϵ-LDP, if for any two private inputs X, X' and any output set $O \subseteq \text{Range}(\mathcal{M})$,*

$$\Pr[\mathcal{M}(X) \in O] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(X') \in O].$$

ϵ-LDP offers homogeneous protection for all input pairs, which may be too "stringent" in some scenarios: No matter how unrelated X and X' are, the output distributions should be statistically similar for small ϵ values, thus rendering the noisy outputs useless.

2.2.1 *Generalization with Distance Metrics.*

To customize heterogeneous privacy guarantees for different pairs of inputs (considering their "actual values"), we resort to LDP on metric spaces [3, 12].

Definition 3. *Let $\epsilon \geq 0$ be the privacy parameter, and d be a suitable distance metric for the input space. $\mathcal{M}$ satisfies ϵd-LDP, if for any two inputs X, X' and any output set $O \subseteq \text{Range}(\mathcal{M})$,*

$$\Pr[\mathcal{M}(X) \in O] \leq e^{\epsilon \cdot d(X,X')} \cdot \Pr[\mathcal{M}(X') \in O].$$

For metric-based LDP, the indistinguishability level of output distributions is now bounded by ϵ times the distance between their respective inputs, and the meaning of ϵ changes for different choices of d. To exploit ϵd-LDP, d needs to be carefully instantiated (see Section 3.2), *e.g.*, L_2-distance for geo-indistinguishability (*i.e.*, different privacy levels within different protection radii).

Interpretation as ϵ-LDP. For a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, its sensitivity w.r.t. a distance metric d on the space $\mathcal{Y}$ is defined as

$$\Delta = \Delta_d f := \max_{\forall X, X' \in \mathcal{X}} d(f(X), f(X')).$$

We use $\mathcal{M} \circ f$ to denote the sequential function execution $\mathcal{M}(f(\cdot))$.

Fact 1. *Let $\mathcal{M}_{\epsilon}$ be an ϵd-LDP mechanism on the space $\mathcal{Y}$ with a metric d, and $f : \mathcal{X} \rightarrow \mathcal{Y}$ be a function with d-sensitivity Δ. Then, the composition $\mathcal{M}_{\epsilon/\Delta} \circ f$ satisfies ϵ-LDP [12, Fact 5].*

(L)DP and its generalization have two desirable properties: *free post-processing* and *composability*. The former means that performing arbitrary computations on the outputs of (L)DP mechanisms incurs no extra privacy loss. The latter enables a modular design of more complicated schemes from basic ones with ϵ as an additive privacy "budget," *e.g.*, sequentially and adaptively running an ϵ-DP mechanism for k times on the same input is at least $k\epsilon$-DP [17].

2.2.2 *Generalized Planar Laplace (PL) Mechanism.*

The PL mechanism [4] is proposed to protect geolocation data in $\mathbb{R}^2$ for ϵd_2-LDP, instantiated with the Euclidean metric d_2. It first draws noise (*i.e.*, two independent random values representing the radius and angle) from a Polar Laplace distribution and then adds the noise back to raw locations in the Cartesian system via a standard transformation.

To perturb data $X \in \mathbb{R}^m$ for $m \geq 2$, the follow-up [60] generalizes the PL mechanism using additive noise $Z \in \mathbb{R}^m$ from distribution

$$\Pr(Z) \propto \exp(\epsilon \cdot ||Z||_2),$$

where $||\cdot||_2$ denotes the L_2-norm. Pragmatically, Z can be obtained by first drawing a uniform vector $Z' \in \mathbb{R}^m$ with $||Z'||_2 = 1$, which is then scaled by a magnitude l from Gamma distribution $\Gamma(m, 1/\epsilon)$.

2.3 Directional Statistics and Distributions

Directional statistics [33] works on vectors' directions, independent of their magnitudes. It can be identified by the universe of m-d unit vectors (for $m \in \mathbb{N}_+$), *i.e.*, all points on the unit $(m-1)$-sphere:

$$\mathbb{S}^{m-1} = \{X \in \mathbb{R}^m : ||X||_2 = 1\}.$$

We consider unimodal distributions $\text{P}_{\mathbb{S}}$ on $\mathbb{S}^{m-1}$ that are rotationally symmetric about a given *mean direction* (or the mode [58]) $\mu \in \mathbb{S}^{m-1}$. To draw vectors from $\text{P}_{\mathbb{S}}$, it is easier to handle marginal distributions obtained by the tangent-normal decomposition; a random vector $Y \in \mathbb{S}^{m-1}$ is decomposed into two components, where one is along μ and the other is along a tangential unit vector $\xi \perp \mu$:

$$Y = t \cdot \mu + \sqrt{1 - t^2} \cdot \xi, \tag{1}$$

where $t = \mu^\top Y$ and ξ is uniformly distributed on subsphere $\mathbb{S}^{m-2} \perp \mu$ due to the rotational symmetry. We can also re-write Eq. (1) as

$$Y = \cos\theta \cdot \mu + \sin\theta \cdot \xi, \tag{2}$$

where $\theta = \arccos(\mu^\top Y)$ is the angular distance between Y and μ. Sampling Y thus boils down to sampling t (resp., θ) from its mixture (resp., angular) density as well as a uniform tangential vector $\xi \perp \mu$.

$\mathsf{P}_{\mathbb{S}}$ has many instances, and we will use the Purkayastha one [58].

Definition 4. *The Purkayastha distribution with mean $\mu \in \mathbb{S}^{m-1}$, parameterized by a concentration value $\kappa \geq 0$, has the density*

$$\mathsf{Pur}(\mu, \kappa)[Y] = C_{\mathsf{Pur}}(m, \kappa) \cdot \exp(-\kappa \cdot \arccos(\mu^\top Y)),$$

where $C_{\mathsf{Pur}}(m, \kappa)$ is the normalization factor: $S_{m-2}^{-1} \cdot F_{m-2,-\kappa}^{-1}(\pi)$ with $S_{m-2} = 2\pi^{\frac{m-1}{2}}\Gamma^{-1}(\frac{m-1}{2})$ and $F_{m-2,-\kappa}(\pi) = \int_0^\pi e^{-\kappa x} \sin^{m-2}(x)\mathrm{d}x$.

The parameter κ specifies how closely Y drawn from $\mathsf{Pur}(\mu, \kappa)$ is "concentrated" about μ: the larger κ, the higher the concentration. $\mathsf{Pur}(\mu, \kappa)$ degenerates to the uniform distribution on $\mathbb{S}^{m-1}$ if $\kappa = 0$.

3 OUR CONSTRUCTIONS

3.1 Overview

Suppose each individual holds a sentence-label pair (X, y) or only X for fine-tuning or testing BERT-based NLP pipelines at an *untrusted* server. Naïvely redacting X (*e.g.*, removing personally identifiable information, PII) is not enough for privacy [51]. We let individuals separately sanitize sentence embedding $\Phi(X)$ extracted by BERT and (sensitive) y for (metric-based) LDP guarantees before being shared. It is in contrast to the conventional central-DP approach [1], which perturbs gradients after centralizing data by a *trusted* curator.

Normalizing $\Phi(X)$ is beneficial for fine-tuning/inference [2]. So, we impose a *consistency* constraint for sanitizing $\Phi(X)$ – the results should also be normalized. We explore two solutions (Section 3.2.1-3.2.2): one is to *directly* draw "replacements" from a distribution defined on a sphere, which can be realized by the Purkayastha mechanism [58] using the angular distance; the other is to *post-process* the noisy embeddings output by the Euclidean-distance-based PL mechanism [60]. However, both suffer from the "curse of dimensionality" since the dimensionality of $\Phi(X)$ is large (*e.g.*, 768). To address it, we add two trainable linear maps between BERT and task layers (Figure 1): one for dimension reduction before adding noise; the other for restoring the dimensionality to maintain utility.

For sanitizing (discrete) labels, we can resort to the randomized response (RR) if the label space is small (which holds for most NLP applications); otherwise, we first prune the label/output space using prior, *e.g.*, obtained by multi-stage training [24] (see Section 3.3).

3.2 Sentence Embeddings Sanitization

In BERT-based NLP pipelines, task layers (typically feed-forward neural networks) are appended to BERT for, *e.g.*, classification. For every input (training/testing) sequence X, a sentence embedding $\Phi(X) \in \mathbb{R}^m$ is extracted by BERT, capturing sentence-level features. Prior arts [2, 66] suggest that normalizing $\Phi(X)$ has many benefits, such as stabilizing training, accelerating convergence, and avoiding overfitting. So, we normalize sentence embeddings to the unit $(m-1)$-sphere[3]: $\Phi(X) \in \mathbb{S}^{m-1}, \forall X$, before inputting them to task layers.

Achieving stringent sequence-level ϵ-LDP may introduce overwhelmingly large noise, detrimental to the downstream task utility. We thus propose to sanitize $\Phi(X)$ for sequence-level ϵd-LDP, which requires us to instantiate a suitable metric d for the input space.

3.2.1 Purkayastha Mechanism. Sentence similarity[4] is typically measured by cosine similarity between sentence embeddings. Yet, cosine similarity is not a suitable distance metric. For any $\Phi(X)$ and $\Phi(X')$ on $\mathbb{S}^{m-1}$, it is thus natural to consider their angular/surface distance, which can be "converted" from cosine similarity:

$$d_\measuredangle(\Phi(X), \Phi(X')) = \arccos(\Phi(X)^\top \cdot \Phi(X')).$$

We then let $d(X, X') = d_\measuredangle(\Phi(X), \Phi(X')), \forall X, X'$, where d satisfies all the axioms of a distance metric since $\Phi(\cdot)$ is injective.

Moreover, we impose a *consistency* constraint: the noisy sentence embeddings $\hat{\Phi}(X)$ should also be on $\mathbb{S}^{m-1}$ to enjoy the normalization benefits. To achieve this, we can *directly* sample "replacements" $\hat{\Phi}(X) \in \mathbb{S}^{m-1}$ from certain $\mathsf{P}_{\mathbb{S}}$ (or take an extra *post-processing* step on noisy outputs detailed in Section 3.2.2). Specifically, we apply the Purkayastha mechanism [58]: sampling $\hat{\Phi}(X)$ from $\mathsf{Pur}(\mu, \kappa)$ with μ as the raw $\Phi(X)$ and κ as the privacy parameter ϵ. It ensures ϵd-LDP for input sequences (see Theorem 1). More importantly, the probability of outputting $\hat{\Phi}(X)$ decreases exponentially with the increasing angular distance between $\hat{\Phi}(X)$ and $\Phi(X)$, which is useful for retaining the utility (as in the exponential mechanism [35]): the shorter the distance $d_\measuredangle(\Phi(X), \Phi(X'))$, the more semantically similar the two sentences X, X' are; hence, the higher the task utility.

Theorem 1. *Given a privacy parameter $\epsilon \geq 0$, our sanitization outputting $\hat{\Phi}(X) \sim \mathsf{Pur}(\Phi(X), \epsilon)$ for $\Phi(X) \in \mathbb{S}^{m-1}$ fulfills $\epsilon d_\measuredangle$-LDP for sentence embeddings (or ϵd-LDP for input sequences).*

The proof is deferred to Appendix A. To efficiently sample $\hat{\Phi}(X) \in \mathbb{R}^m$ (*e.g.*, $m = 768$ for BERT-base), we exploit the tangent-normal decomposition in Eq. (2). The key step is to draw θ from its density

$$\mathsf{PurArc}(m, \epsilon)[\theta] = F_{m-2,-\epsilon}^{-1}(\pi) \cdot \sin^{m-2}(\theta)e^{-\epsilon\theta},$$

where $F_{m-2,-\epsilon}(\pi) = \int_0^\pi \sin^{m-2}(\theta)e^{-\epsilon\theta}$. This can be done using an approximate inversion method [58, Algorithm 1]. We then need to draw a tangential unit vector ξ uniformly from $\mathbb{S}^{m-2} \perp \Phi(X)$: In practice, we first sample a random m-dimensional vector ξ (via the standard normal distribution), then make it orthogonal to $\Phi(X)$:

$$\xi = \xi - (\Phi(X)^\top \cdot \xi) \times \Phi(X),$$

and normalize it to a unit one. Finally, the noisy replacement is

$$\hat{\Phi}(X) = \cos(\theta) \cdot \Phi(X) + \sin(\theta) \cdot \xi.$$

Empirically, we find that the noise "magnitude" $\theta \sim \mathsf{PurArc}(m, \epsilon)$ increases with m for fixed ϵ, incurring the "curse of dimensionality." The prior work only evaluates the feasibility of the Purkayastha mechanism on spatial and temporal data with $m = 2$ or 3 [58]. For our big m (*e.g.*, 768 for BERT-Base), our pilot results show that task models cannot converge even under a low privacy regime ($\epsilon = 50$).

[3] A sphere $r\mathbb{S}^{m-1}$ of radius $r > 1$ or $r < 1$ yields similar performance.

[4] https://huggingface.co/tasks/sentence-similarity

To escape from the curse, we introduce two linear maps $M_1, M_2 \in \mathbb{R}^{m' \times m}$ (with $m' \ll m$) between BERT and task layers. They are randomly initialized (hence not necessarily inverse of each other) and updated using gradients (like pipeline weights). We first use M_1 to transform each raw sentence embedding to $M_1 \cdot \Phi(X)$ and then generate a replacement with smaller $\theta' \sim \mathsf{PurArc}(m', \epsilon)$. M_1 mimics random projection [60] to reduce dimension while approximately preserving the raw pairwise distances. Our privacy is not affected since the proof is dimension-independent. Meanwhile, our sampling efficiency can be remarkably improved due to a much smaller m'. Finally, we project each replacement back to $\mathbb{S}^{m-1}$ via M_2 (no extra privacy loss due to the free post-processing). With M_1 and M_2, we almost hit the non-private task accuracy (to be shown in Section 4).

3.2.2 Normalized PL Mechanism. Since $\mathbb{S}^{m-1}$ is still in a Euclidean space, it is also meaningful to consider d as the Euclidean distance d_2

$$d(X, X') = d_2(\Phi(X), \Phi(X')) = ||\Phi(X) - \Phi(X')||_2.$$

To ensure *sequence-level* ϵd_2-LDP (which we formally assert below), one can run the generalized PL mechanism [60] to perturb sentence embeddings $\Phi(X)$ by using additive noise $Z \in \mathbb{R}^m$ as

$$\hat{\Phi}(X) = \Phi(X) + Z, \text{ with } Z \propto \exp(\epsilon \cdot ||Z||_2).$$

Theorem 2. *Given a privacy parameter $\epsilon \geq 0$, sanitizing $\Phi(X)$ by the generalized PL mechanism satisfies sequence-level ϵd_2-LDP.*

The proof is similar to that of Theorem 1, which we omit here. We note that prior arts [23, 44] also used the generalized PL mechanism to perturb the embeddings of every token (in a sequence) but only for *token-level* ϵd_2-LDP, which is weaker than our sequence-level notion. Theoretically, upgradation for sentence-level LDP requires scaling the privacy bound by the sequence length as $(n \cdot \epsilon d_2)$-LDP. Besides, they focus on simpler tasks than ours, *e.g.*, mapping noisy token embeddings back to text by nearest neighbor search [23].

Corollary 1. *For any $\Phi(X), \Phi(X') \in \mathbb{S}^{m-1}$, $d_2(\Phi(X), \Phi(X')) \leq d_\measuredangle(\Phi(X), \Phi(X'))$, so our PL-based sanitization is also $\epsilon d_\measuredangle$-LDP.*

The noise magnitude drawn from $\Gamma(m, 1/\epsilon)$ is also scaled by m, which can be reduced properly to m' by the linear transformation. However, the task utility may still be undesirable since the noisy embeddings are in the entire Euclidean space $\mathbb{R}^{m'}$ rather than $\mathbb{S}^{m-1}$, *i.e.*, the *consistency* is violated. As a remedy, we *post-process* $\hat{\Phi}(X)$ by normalizing it to $\mathbb{S}^{m-1}$ without hurting our privacy guarantees, leading to our normalized PL mechanism with prior on the L_2-norm.

3.3 Fine-tuning/Testing with(out) Label Privacy

We consider that an untrusted server starts fine-tuning from a raw, public BERT checkpoint $\Phi(\cdot)$. In each fine-tuning step, the server chooses a user *batch* (of tunable size) to provide the latest $\Phi(\cdot)$; the valuable task layers are *never* disclosed to the users. Each user in the batch can compute its sentence embedding $\Phi(X)$, which is sanitized by using either the Purkayastha or normalized PL mechanism (in Section 3.2) to $\hat{\Phi}(X)$ for metric-LDP and then shared with the server.

Given the pairs of $(\hat{\Phi}(X), y)$ for a batch, when the label y is non-sensitive (*e.g.*, just a single bit denoting positive/negative), the server can then fine-tune *entire* pipelines without accessing the raw sentences X; it lets the gradient back-propagate to update all the parameters for better performance, albeit $\Phi(\cdot)$ could be frozen.

Typically, an epoch refers to an entire transit of a training dataset through the pipeline, *i.e.*, every X is used only once per epoch. The number of epochs k is a hyperparameter; we need to estimate the overall privacy loss for k times sanitization on the same X. Given the basic composition theorem [17], we have at least $(k\epsilon)d$-LDP for each user. Such a bound is almost optimal for small k (*e.g.*, inference); otherwise, one could derive a tighter bound via tailored privacy accounting tools such as Rényi DP [37]. We leave it as future work.

3.3.1 Sanitizing Labels for LDP. When labels are deemed sensitive, such as the PAC setting [13] and online advertising [24], we should also sanitize them by a mechanism $\mathcal{M}_l$ to preserve label privacy. Formally, we propose label-LDP as a local version of label-DP [24].

Definition 5. *Given a privacy parameter $\epsilon \geq 0$, $\mathcal{M}_l$ is ϵ-label-LDP, if for any two labels y, y' and any output set $O \subseteq \mathsf{Range}(\mathcal{M}_l)$,*

$$\Pr[\mathcal{M}_l(y) \in O] \leq e^\epsilon \cdot \Pr[\mathcal{M}_l(y') \in O].$$

For most NLP applications, *e.g.*, bi-/multi-nary classification in the GLUE benchmark [54], the size $|\mathbf{y}|$ of (discrete) label space is often small. A simple yet effective instantiation of $\mathcal{M}_l$ for discrete data is randomized response (RR) [56] proposed decades ago. Concretely, for a privacy parameter $\epsilon \geq 0$ and label space $\mathbf{y}$, RR perturbs a true label y to itself $\hat{y} = y$ with the probability

$$\Pr[y = \hat{y}] = e^\epsilon / (e^\epsilon + |\mathbf{y}| - 1),$$

or to $\forall \hat{y} \in \mathbf{y} \setminus y$ uniformly. With RR, we achieve ϵ-LDP for labels.

When $|\mathbf{y}|$ is large, we employ the adapted RR [24], which exploits a prior distribution $\mathbf{p}$ to prune the label space $\mathbf{y}$ to a smaller one $\mathbf{y}'$. The prior $\mathbf{p}$ can be obtained publicly, *e.g.*, auxiliary labeled corpora similar to the assembly of users' data. If it is not available, one can first bootstrap it from a uniform one and then progressively refine it (by the previous round model outputs) via multi-stage training [24]. With $\mathbf{p}$, one can estimate an optimal $|\mathbf{y}'|$ – labels with top-$|\mathbf{y}'|$ prior probabilities to maximize the signal-to-noise ratio $\Pr[y = (\hat{y} = y)]$. The adapted RR with prior also ensures ϵ-label-LDP.

Theorem 3. *Let $\epsilon_1, \epsilon_2 \geq 0$ be two privacy parameters, $\mathcal{M}$ be the Purkayastha mechanism, and $\mathcal{M}_l$ be the (adapted) RR. Sanitizing $\Phi(X)$ and y respectively by $\mathcal{M}$ and $\mathcal{M}_l$ satisfies $(\epsilon_1 d_\measuredangle + \epsilon_2)$-LDP.*

Due to the space issue, the proof is deferred to Appendix A.

3.3.2 Inference at the Server. With fine-tuned $\Phi(\cdot)$, users only need to sanitize their *test* sentence embeddings for inference. Aligning to the noisy fine-tuning is beneficial for inference accuracy and can also mitigate embedding-based attacks [41, 48] on test sequences.

Local inference (without any DP noise), as in DP-SGD, forces the server to reveal its entire pipelines, losing its intellectual property and incurring more user-side (time and storage) overheads.

3.4 Privacy Amplification by Shuffling

DP-SGD fine-tuning and our approaches with label privacy consider different privacy models (approximate CDP vs. metric-LDP), which are not directly comparable. The shuffle model is an "intermediate" trust model [19] that gained significant interest for bringing the best of both central and local models. It relies on a third party to anonymize (or randomly shuffle) users' sanitized embedding-label pairs before being sent to the server. The anonymity "amplifies" privacy without any extra noise addition, allowing us to claim much

	SST-2	IMDb	QNLI
#train samples	67, 349	25, 000	104, 743
#test samples	872	25, 000	5, 463

Table 1: Statistics of the task datasets

stronger privacy guarantees of our metric-LDP approaches when seen in the central model, with a $\Theta(\sqrt{N})$ amplification factor for a total of N users [19]. Tighter results are derived [21, 22] with a new reduction analysis of Rényi DP parameters for the shuffled outputs.

For fair comparisons with DP-SGD under the shuffle model, we should sanitize sentence embeddings by Gaussian noise [1] to eliminate the metric factor induced by the Purkayastha or generalized PL mechanism. It would also be interesting to explore adding Gaussian noise to the hidden (rather than only sentence) embeddings output by different layers inside LMs. We leave them as future work.

4 EXPERIMENTS

4.1 Experimental Setup

Our experiments are run over a cluster comprising Tesla P100 GPUs. We implemented both Purkayastha and normalized planar Laplace mechanisms for sanitizing sentence embeddings, as well as RR for sanitizing labels using Python. We adopted `BERT-base-uncased`[5] (from the Huggingface Transformers library in PyTorch) as our LM checkpoint for fine-tuning given sanitized embedding-label pairs. We also consider the non-private baseline without any DP noise.

Datasets. We employ three representative tasks that have been used in differentially-private NLP [23, 29, 63–65]: i) Stanford sentiment treebank (SST-2) [54], ii) Internet movie database (IMDb) [32], and iii) Question-answering natural language inference (QNLI) [54]. All are web-related: The first two are for positive/negative sentiment classification of online movie reviews; the last one is to check if the context distilled from Wikipedia contains the answer to a question. They also have privacy risks, *e.g.*, the authorship can be identified by stylistic features like word frequencies; our approaches can be a general remedy. We use their dev sets as the test sets since the "real" ones are missing. Table 1 shows the size of training and dev sets.

In the following experiments, all the *test* sentence embeddings are sanitized for alignment to those in fine-tuning. The utility metric is *accuracy* w.r.t. the ground-truth labels of test sequences.

Hyperparameters. For all the tasks, we set the number of epochs as 3, learning rate as 2×10^{-5}, and batch size as 64. We keep others (*e.g.*, no weight and learning rate decay) default as literature [54].

4.2 Configuring Linear Map Dimensions

As m is fixed as 768 in BERT-Base [15], m' is the only tunable parameter of linear maps affecting performance. By setting $\epsilon_1 = 10$ for metric-LDP, Table 2 shows the task accuracy of the two sanitization mechanisms for sentence embeddings and the non-private baseline when tuning m' (without label privacy). The last line ($m' = 768$) is the setting without dimension reduction. The noise scale decreases as m' decreases; the accuracy of both two sanitization mechanisms increases since the signal-to-noise becomes larger for a unit sphere. Efficiency-wise, the noise sampling rates are also faster for smaller m'. Nevertheless, the total fine-tuning time, dominated by matrix computations in pipelines, keeps almost unchanged compared to the non-private baseline (*e.g.*, ~74 seconds for SST-2). While, DP-SGD fine-tuning is 10–100× slower than the non-private baseline [9].

Cautions. For most applications (*e.g.*, multi/binary classification in our experiments), such dimension reduction-then-ascension has a limited impact on task accuracy without DP noise. But for a few much more complicated tasks, we observe notable accuracy drops when m' is very small (*e.g.*, <10), probably because low-dimension vectors cannot encode sufficient information for downstream tasks. To balance everything, we set $m' = 16$, which yields the best accuracy comparable to the non-private baseline, for later experiments.

4.3 Task Accuracy with(out) Label Privacy

We first consider that the labels in all the tasks do not need protection. For sequence-level LDP, the prior art [36] suggests that $\epsilon < 10$ indicates a strong privacy regime, $10 \leq \epsilon < 20$ is moderate privacy, and $\epsilon \geq 20$ is seen as weak privacy; so we tune our privacy parameter ϵ_1 from 1 to 12 for metric-LDP with d_2 or $d_{\measuredangle}$. The results of all three tasks are shown in Figure 2. The larger ϵ_1 leads to better accuracy of both approaches. At a strong privacy regime (*e.g.*, $\epsilon_1 \leq 4$), the normalized Laplace mechanism outperforms the Purkayastha one, *e.g.*, by as much as 0.14 for QNLI; the Purkayastha one will be slightly better for $\epsilon_1 > 8$. Both designs can almost hit the non-private accuracy (*e.g.*, 0.93 for SST-2 at $\epsilon_1 = 12$), much better than simply "upgrading" token-level designs [23, 65] by $n = 128$.

We then consider sanitizing labels to fully protect *every* individual's training example (*i.e.*, a sequence-label pair, vs. DP-SGD offers example-level CDP). Since all the tasks are binary classifications, we apply the randomized response (RR) for each label independently (without pruning the label space by any prior). We tune ϵ_2 from 0.5 to 3 for label-LDP while fixing $\epsilon_1 = 8$ for sequence-level metric-LDP. The accuracy of all the tasks fine-tuned on sanitized sequence-label pairs is shown in Figure 3, which only reduces by 0.01 to 0.19 with small budgets for extra label privacy. A very recent work [63] reports that DP-SGD achieves an averaged accuracy of 0.685 on four tasks (including SST-2 and QNLI) using $\epsilon = 6.7$ for example-level *central* DP. Our approaches offer a promising direction to fine-tune more accurate models while ensuring stronger LDP guarantees.

4.4 Defenses Against Embedding-based Attacks

As the recent taxonomy of attacks on embeddings [48], we consider MIAs, embedding inversion, and sensitive attribute inference. Adversarially tampering models (*e.g.*, poisoning) is outside our scope.

MIAs. They often exploit that models may behave differently on the training data versus never-before-seen data [10, 62]. We consider *sequence-level* MIAs – whether a target sequence is in the sequence ensemble. With targets, the adversary can issue queries to the *black-box* fine-tuned pipeline for the prediction confidences/probabilities. We employ two simple yet effective threshold-based attacks [49, 62], comparable to the shadow-learning attack [47]. One treats those with confidences larger than a threshold τ as training data [62] since the confidence of predicting a training sequence as its truth label should be close to 1. Similarly, the other exploits that the entropy of prediction for a training sequence should be close to 0, which can be further improved by encoding the truth labels [49].

[5] https://huggingface.co/bert-base-uncased

Reduced size m'	SST-2			IMDb			QNLI		
	PurMech	LapMech	Non-DP	PurMech	LapMech	Non-DP	PurMech	LapMech	Non-DP
16	**0.9243**	0.9209	0.9335	0.8777	**0.8792**	0.8912	**0.9081**	0.9046	0.9083
32	**0.8853**	0.8830	0.9243	0.8419	**0.8511**	0.8892	**0.8770**	0.8752	0.9081
64	0.8050	**0.8326**	0.9266	0.7827	**0.7974**	0.8878	0.8058	**0.8237**	0.9112
128	0.7534	**0.7557**	0.9243	0.7188	**0.7358**	0.8880	0.7230	**0.7557**	0.9099
256	0.6663	**0.6789**	0.9226	0.6350	**0.6989**	0.8895	0.6421	**0.6837**	0.9101
512	0.5103	**0.6296**	0.9289	0.5920	**0.6492**	0.8891	0.5056	**0.6262**	0.9134
768	0.5087	**0.5524**	0.9207	0.5532	**0.5738**	0.8905	0.5037	**0.5842**	0.9125

Table 2: Accuracy of the two mechanisms and non-DP baseline when tuning the dimension m' with $\epsilon_1 = 10$ for metric-LDP

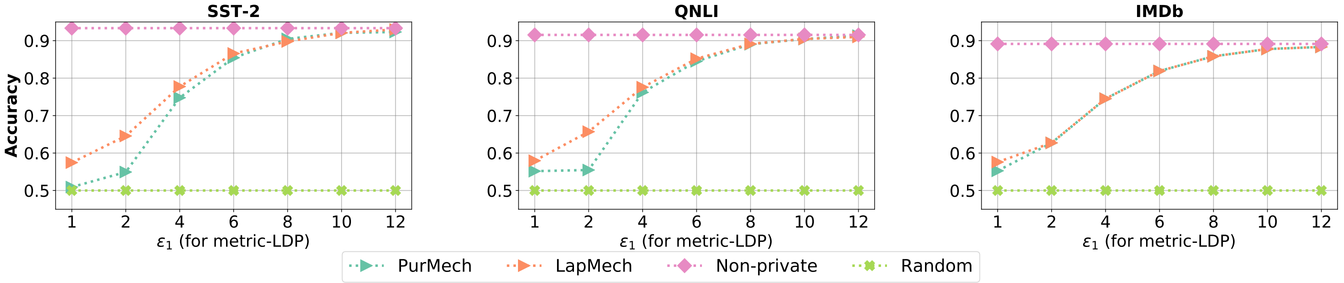

Figure 2: Accuracy on sanitized sentence embeddings when tuning ϵ_1 for metric-LDP without label privacy

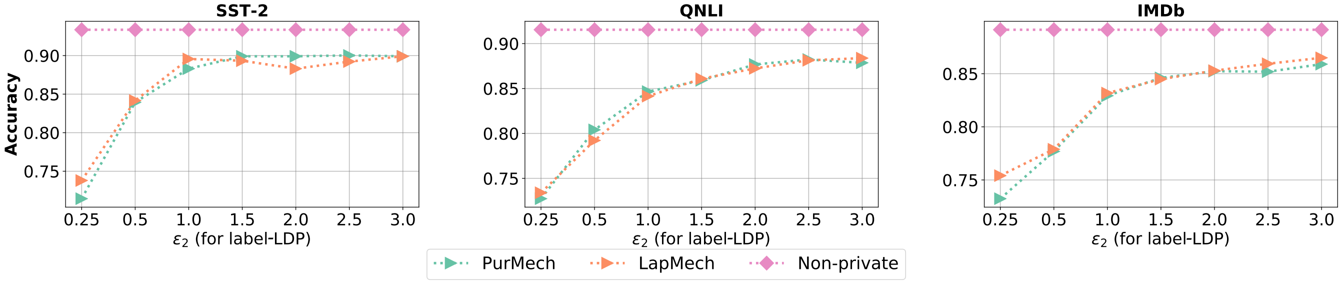

Figure 3: Accuracy on sanitized embedding-label pairs when tuning ϵ_2 for label-LDP (with $\epsilon_1 = 8$)

ϵ_1	Entropy		Confidence	
	PurMech	LapMech	PurMech	LapMech
1	0.502	0.496	0.503	0.508
4	0.519	0.518	0.508	0.509
8	0.536	0.538	0.511	0.513
12	0.551	0.552	0.523	0.522
∞	0.659		0.645	

Table 3: Success rates of two MIAs on SST-2

We follow the pre-processing [64]: evenly splitting a dataset into two subsets, one for setting τ and the other for reporting the success rates. For easier attacks, we use extra tricks [59] (*e.g.*, random drops of tokens) to increase (resp. decrease) the prediction confidence (resp. entropy) of training sequences. We evaluated the two MIAs on SST-2 (without label privacy) and summarize the results in Table 3. For all choices of ϵ, the success rates of two MIAs are reduced to ~0.5 (or random guessing), much smaller than the non-DP baselines.

Embedding Inversion Attacks. The attacks try to recover (unordered) tokens $\{x_i\}_{i\in[n]} \subseteq X$, *e.g.*, identity numbers [41], from the sanitized sentence embedding $\hat{\Phi}(X)$. We use a two-step attack [48], assuming a *white-box* adversary with the knowledge of the pipeline weights and architecture (stronger than its black-box version [48]). It maps $\hat{\Phi}(X)$ to a lower-layer embedding by a learned mapping (*e.g.*, linear least square models) and then selects a set of tokens X^* to minimize the L_2-distance between its lower-layer embedding and the mapped one of $\hat{\Phi}(X)$. Embeddings from deeper layers are more "abstract" and harder to be inverted; *e.g.*, the recovery rates (even without DP noise) were shown to approach 0 [48, Figure 2].

Attribute Inference Attacks. Apart from recovering exact tokens, sensitive attributes (such as text authorship) that are inherent in X (and independent of training objectives) may be inferred from $\hat{\Phi}(X)$. As [48], we consider a *black-box* adversary who can collect a limited, auxiliary dataset $\mathcal{D}_{aux}$ of sequences labeled with sensitive attributes. The set of all possible sensitive attributes (*e.g.*, authors) of interest is also known. The adversary can then treat the inference as a downstream task: It trains a classifier on $\mathcal{D}_{aux}$ (like the noisy fine-tuning in Section 3.3), which is used for prediction on $\hat{\Phi}(X)$.

We consider IMDb as $\mathcal{D}_{aux}$ with the film genres being sensitive attributes. For the classifier, we train a three-layer neural network to infer the film genres from sanitized embeddings of movie reviews in SST-2. Table 4 shows that our two sanitization mechanisms can "transform" the inference to a majority-class case that assigns all of

	action	comedy	drama	horror	Overall
Non-DP	0.78	0.836	0.404	0.455	0.659
PurMech ($\epsilon_1 = 8$)	1.0	**0**	**0**	**0**	**0.276**
LapMech ($\epsilon_1 = 8$)	1.0	**0**	**0**	**0**	**0.276**

Table 4: Success rates of sensitive attribute inference

the labels to the majority class ('action') in the target data, reducing the overall success rate by ~0.42 compared to the non-DP baseline.

5 RELATED WORK

5.1 Privacy Risks in NLP

Song and Raghunathan [48] taxonomize embedding-based attacks into three categories, covering a wider scope than their concurrent work [41]: inverting partial raw text, inferring sensitive attributes (*e.g.*, authorship, gender) and membership (*i.e.*, the is-in relationship between victims and private training data). Several others [9, 10] study training data "memorization" (a.k.a. membership inference) given only black-box access to generative LMs. Such memorization can even lead to more devastating attacks, *e.g.*, extracting verbatim training text [10]. Beguelin *et al.* [6] introduce differential scores and rank for analyzing the "update" leakage to recover the new data for updating/fine-tuning LMs. Incorporating DP (for generating embeddings or training) to thwart these risks is thus vital. Model extraction and active attacks (*e.g.*, poisoning) are out of our scope.

5.2 Privacy-preserving Text Embeddings

Two lines of work study generating privacy-preserving text embeddings; one (including ours) resorts to DP, and the other is through adversarial training. SynTF [57] synthesizes term-frequency (TF) vectors by using the exponential mechanism (EM) [35] to sample a replacement for each raw term in a document. However, TF vectors capturing document-level statistics have limited applications (*e.g.*, simple text mining tasks). Feyisetan *et al.* [23] apply the generalized PL mechanism [60] to perturb non-contextualized *token embeddings* for metric-LDP instantiated by d_2 and further post-process them to sanitized text [65] by nearest-neighbor search. Their follow-up [44] can get noisy sequence representations but still from the token-wise perturbation [23]. Instead, Lyu *et al.* [31] directly perturb BERT-based sentence embeddings by the Laplace mechanism [17] for pure LDP. All these works protect token/sentence embeddings only at a *weaker* token level. A very recent work [36] incorporates EM to sample a replacement of a document embedding, as an average of all sentence embeddings, to ensure sentence-level privacy: hiding the impact of any single sentence in a document. Yet, its core is central DP, like DP-SGD. It also requires dedicated efforts to prepare a candidate set of non-private document embeddings used in EM; our "candidate" space can be the entire unit $(m-1)$-sphere.

For adversarial-training-based schemes [14, 18, 30], a simulated adversary is trained to infer any sensitive information jointly with a main model that tries to maximize the adversary's loss and minimize the primary learning objective. The learned private representations can effectively mitigate inference-time attacks [48], but they are not general-purpose like DP ones since the learning goal is task-specific.

5.3 Training NLP Models with DP

DP-SGD [1] modifies the mini-batch stochastic optimization process by adding Gaussian noise to aggregated gradients in each training step such that the final models are DP. An early attempt [34] trains LSTM-based LMs by deploying DP-SGD in the federated learning setting. By setting hyperparameters properly (*e.g.*, mega-batch sizes) and using DP-SGD with Adam optimizer, one can even pre-train gigantic LMs privately but requires Google TPUs [5]. Yu *et al.* [64] propose reparametrized gradient perturbation (RGP): It can perturb "dimension-reduced" gradients with less noise, but it is costly due to the reparameterization in every update and makes training unstable. The follow-up [63] addresses these two issues by deriving a small number of new parameters (from, *e.g.*, LoRA and Compacter) that can be "plugged in" frozen LMs and running DP-SGD on them without hurting much performance. Li *et al.* [29] propose a memory-saving technique: ghost clipping, which allows running DP-SGD for full fine-tuning without storing per-example gradients (as large as LMs). Fine-tuning based on DP-SGD (or its more efficient variants) is in great contrast to ours: perturbing larger gradients in back-propagation (vs. smaller sentence embeddings in forward pass) and offering central DP with a trusted party (vs. LDP without any trust).

Label-only DP is formally introduced by Chaudhuri and Hsu [13] for private PAC-learners, later considered in deep learning [20, 24]. Ghazi *et al.* [24] perturb discrete labels by the adapted RR that uses prior to prune the label space. The prior may be publicly available as domain knowledge or obtained from multi-stage training. A parallel work [20] proposes two designs based on the Laplace mechanism with Bayesian inference and the PATE framework [42]. Yet, label-only DP has pitfalls: an attacker can de-noise perturbed labels [8].

6 CONCLUSION

The web is a text-centric environment, collecting text inputs and providing large pre-trained LM-based NLP applications, *e.g.*, ChatGPT. Their success requires gathering collective intelligence, but severe privacy risks may hinder individual involvement. Sanitizing text data locally to protect privacy thus becomes significant.

Naïvely removing common PII in text is not enough. Processing tokens/words or their embeddings independently loses context. We thus sanitize sentence embeddings that encode contextual information. We build two sanitization approaches atop the Purkayastha and generalized planar Laplace mechanisms, ensuring metric-LDP. They can be integrated to modern LM-based NLP pipelines working on (noisy) sentence embeddings. For better utility, we normalize embeddings for consistency and mitigate the "curse of dimensionality" by strategic uses two extra trainable linear maps. To fully protect training data, we also sanitize labels by the classic random response.

We conduct extensive experiments on three representative NLP tasks. We also empirically evaluate how the linear-map size impacts task accuracy. The results confirm that our sanitization approaches are efficient, accurate, and effective in thwarting various privacy attacks in practice. Altogether, our new perspective leads to a better approach to deep neural network training with DP, challenging the traditional wisdom perturbing gradients. As a new paradigm to ensure LDP in both training and inference, there are many promising future directions for privacy-preserving deep learning research, *e.g.*, generalizing to (transformer-based) computer vision tasks.

ACKNOWLEDGMENTS

Sherman Chow is supported in parts by the General Research Funds (CUHK 14209918, 14210319, 14210621), University Grants Committee, Hong Kong. Authors at OSU are sponsored in part by NSF CAREER #1942980, and Ohio Supercomputer Center [40].

REFERENCES

[1] Martín Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. In *CCS*. 308–318.

[2] Prince Osei Aboagye, Yan Zheng, Chin-Chia Michael Yeh, Junpeng Wang, Wei Zhang, Liang Wang, Hao Yang, and Jeff M. Phillips. 2022. Normalization of Language Embeddings for Cross-Lingual Alignment. In *ICLR*. 32 pages.

[3] Mário S. Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. 2018. Local Differential Privacy on Metric Spaces: Optimizing the Trade-Off with Utility. In *CSF*. 262–267.

[4] Miguel E. Andrés, Nicolás Emilio Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. 2013. Geo-indistinguishability: differential privacy for location-based systems. In *CCS*. 901–914.

[5] Rohan Anil, Badih Ghazi, Vineet Gupta, Ravi Kumar, and Pasin Manurangsi. 2022. Large-Scale Differentially Private BERT. In *Findings of EMNLP*. 6481–6491.

[6] Santiago Zanella Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. 2020. Analyzing Information Leakage of Updates to Natural Language Models. In *CCS*. 363–375.

[7] Jonas Böhler and Florian Kerschbaum. 2021. Secure Multi-party Computation of Differentially Private Heavy Hitters. In *CCS*. 2361–2377.

[8] Robert Istvan Busa-Fekete, Andres Munoz Medina, Umar Syed, and Sergei Vassilvitskii. 2021. On the pitfalls of label differential privacy. In *NeurIPS Workshop*. 6 pages.

[9] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks. In *USENIX Security*. 267–284.

[10] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom B. Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting Training Data from Large Language Models. In *USENIX Security*. 2633–2650.

[11] Melissa Chase and Sherman S. M. Chow. 2009. Improving privacy and security in multi-authority attribute-based encryption. In *CCS*. 121–130.

[12] Konstantinos Chatzikokolakis, Miguel E. Andrés, Nicolás Emilio Bordenabe, and Catuscia Palamidessi. 2013. Broadening the Scope of Differential Privacy Using Metrics. In *PETS*. 82–102.

[13] Kamalika Chaudhuri and Daniel J. Hsu. 2011. Sample Complexity Bounds for Differentially Private Learning. In *COLT*. 155–186.

[14] Maximin Coavoux, Shashi Narayan, and Shay B. Cohen. 2018. Privacy-preserving Neural Representations of Text. In *EMNLP*. 1–10.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[16] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *TCC*. 265–284.

[17] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* 9, 3-4 (2014), 211–407.

[18] Yanai Elazar and Yoav Goldberg. 2018. Adversarial Removal of Demographic Attributes from Text Data. In *EMNLP*. 11–21.

[19] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *SODA*. 2468–2479.

[20] Mani Malek Esmaeili, Ilya Mironov, Karthik Prasad, Igor Shilov, and Florian Tramèr. 2021. Antipodes of Label Differential Privacy: PATE and ALIBI. In *NeurIPS*. 6934–6945.

[21] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2021. Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling. In *FOCS*. 954–964.

[22] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2022. Stronger Privacy Amplification by Shuffling for Rényi and Approximate Differential Privacy. arXiv:2208.04591.

[23] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2020. Privacy- and Utility-Preserving Textual Analysis via Calibrated Multivariate Perturbations. In *WSDM*. 178–186.

[24] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. 2021. Deep Learning with Label Differential Privacy. In *NeurIPS*. 27131–27145.

[25] Thomas Humphries, Rasoul Akhavan Mahdavi, Shannon Veitch, and Florian Kerschbaum. 2022. Selective MPC: Distributed Computation of Differentially Private Key-Value Statistics. In *CCS*. 1459–1472.

[26] Peter Kairouz, Kallista A. Bonawitz, and Daniel Ramage. 2016. Discrete Distribution Estimation under Local Privacy. In *ICML*. 2436–2444.

[27] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. 2008. What Can We Learn Privately?. In *FOCS*. 531–540.

[28] Eric Lehman, Sarthak Jain, Karl Pichotta, Yoav Goldberg, and Byron C. Wallace. 2021. Does BERT Pretrained on Clinical Notes Reveal Sensitive Data?. In *NAACL-HLT*. 946–959.

[29] Xuechen Li, Florian Tramèr, Percy Liang, and Tatsunori Hashimoto. 2022. Large Language Models Can Be Strong Differentially Private Learners. In *ICLR*. 30 pages.

[30] Yitong Li, Timothy Baldwin, and Trevor Cohn. 2018. Towards Robust and Privacy-preserving Text Representations. In *ACL*. 25–30.

[31] Lingjuan Lyu, Xuanli He, and Yitong Li. 2020. Differentially Private Representation for NLP: Formal Guarantee and An Empirical Study on Privacy and Fairness. In *Findings of EMNLP*. 2355–2365.

[32] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *ACL*. 142–150.

[33] Kanti V Mardia and Peter E Jupp. 2000. *Directional statistics.*

[34] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. 2018. Learning Differentially Private Recurrent Language Models. In *ICLR*. 14 pages.

[35] Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy. In *FOCS*. 94–103.

[36] Casey Meehan, Khalil Mrini, and Kamalika Chaudhuri. 2022. Sentence-level Privacy for Document Embeddings. In *ACL*. 3367–3380.

[37] Ilya Mironov. 2017. Rényi Differential Privacy. In *CSF*. 263–275.

[38] Moni Naor, Benny Pinkas, and Omer Reingold. 1999. Distributed Pseudo-random Functions and KDCs. In *EUROCRYPT*. 327–346.

[39] Lucien K. L. Ng and Sherman S. M. Chow. 2021. GForce: GPU-Friendly Oblivious and Rapid Neural Network Inference. In *USENIX Security*. 2147–2164.

[40] OSC. 1987. Ohio Supercomputer Center. http://osc.edu/ark:/19495/f5s1ph73

[41] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. 2020. Privacy Risks of General-Purpose Language Models. In *S&P*. 1314–1331.

[42] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. 2017. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In *ICLR*. 16 pages.

[43] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.

[44] Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. 2021. Natural Language Understanding with Privacy-Preserving BERT. In *CIKM*. 1488–1497.

[45] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. OpenAI Report.

[46] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP-IJCNLP*. 3980–3990.

[47] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership Inference Attacks Against Machine Learning Models. In *S&P*. 3–18.

[48] Congzheng Song and Ananth Raghunathan. 2020. Information Leakage in Embedding Models. In *CCS*. 377–390.

[49] Liwei Song and Prateek Mittal. 2021. Systematic Evaluation of Privacy Risks of Machine Learning Models. In *USENIX Security*. 2615–2632.

[50] Timothy Stevens, Christian Skalka, Christelle Vincent, John Ring, Samuel Clark, and Joseph Near. 2022. Efficient Differentially Private Secure Aggregation for Federated Learning via Hardness of Learning with Errors. In *USENIX Security*. 1379 – 1395.

[51] Latanya Sweeney. 2015. Only You, Your Doctor, and Many Others May Know. *Technology Science* 2015092903, 9 (2015), 29.

[52] Sijun Tan, Brian Knott, Yuan Tian, and David J. Wu. 2021. CryptGPU: Fast Privacy-Preserving Machine Learning on the GPU. In *S&P*. 1021–1038.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS*. 5998–6008.

[54] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *ICLR*. 20 pages. Datasets are available at https://gluebenchmark.com/tasks.

[55] Tianhao Wang, Milan Lopuhaä-Zwakenberg, Zitao Li, Boris Skoric, and Ninghui Li. 2020. Locally Differentially Private Frequency Estimation with Consistency. In *NDSS*. 16 pages.

[56] Stanley L Warner. 1965. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *JASA* 60, 309 (1965), 63–69.

[57] Benjamin Weggenmann and Florian Kerschbaum. 2018. SynTF: Synthetic and Differentially Private Term Frequency Vectors for Privacy-Preserving Text Mining. In *SIGIR*. 305–314.

[58] Benjamin Weggenmann and Florian Kerschbaum. 2021. Differential Privacy for Directional Data. In *CCS*. 1205–1222.

[59] Jason W. Wei and Kai Zou. 2019. EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks. In *EMNLP-IJCNLP*. 6381–6387.

[60] Xi Wu, Fengan Li, Arun Kumar, Kamalika Chaudhuri, Somesh Jha, and Jeffrey F. Naughton. 2017. Bolt-on Differential Privacy for Scalable Stochastic Gradient Descent-based Analytics. In *SIGMOD*. 1307–1322.

[61] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. arXiv:1609.08144.

[62] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting. In *CSF*. 268 – 282.

[63] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A. Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. 2022. Differentially Private Fine-tuning of Language Models. In *ICLR*. 19 pages.

[64] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. 2021. Large Scale Private Learning via Low-rank Reparametrization. In *ICML*. 12208–12218.

[65] Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. 2021. Differential Privacy for Text Analytics via Natural Text Sanitization. In *Findings of ACL/IJCNLP*. 3853–3866.

[66] Wenxuan Zhou, Junyi Du, and Xiang Ren. 2019. Improving BERT Fine-tuning with Embedding Normalization. arXiv:1911.03918.

A MISSING PROOFS IN SECTION 3

Proof of Theorem 1. Let X, X' be any two sequences. The embeddings are $\Phi(X)$ and $\Phi(X')$. For any possible output $Y \in \mathbb{S}^{m-1}$,

$$\begin{aligned}\frac{\mathsf{Pur}(\Phi(X),\epsilon)[Y]}{\mathsf{Pur}(\Phi(X'),\epsilon)[Y]} &= \frac{C_{\mathsf{Pur}} \cdot \exp(-\epsilon \cdot \arccos(\Phi(X)^{\top} \cdot Y))}{C_{\mathsf{Pur}} \cdot \exp(-\epsilon \cdot \arccos(\Phi(X')^{\top} \cdot Y))} \\ &= \frac{\exp(-\epsilon \cdot d_{\measuredangle}(\Phi(X), Y))}{\exp(-\epsilon \cdot d_{\measuredangle}(\Phi(X'), Y))} \\ &= \exp(\epsilon \cdot (d_{\measuredangle}(\Phi(X), Y) - d_{\measuredangle}(\Phi(X'), Y))) \\ &\leq \exp(\epsilon \cdot d_{\measuredangle}(\Phi(X), \Phi(X'))) \\ &= \exp(\epsilon \cdot d(X, X')).\end{aligned}$$

After canceling out C_{Pur}, we can apply the triangle inequality of $d_{\measuredangle}$. The last step is due to $d(X, X') = d_{\measuredangle}(\Phi(X), \Phi(X'))$. □

Proof of Theorem 3. Given any two sentence embedding-label pairs $(\Phi(X), y)$ and $(\Phi(X'), y')$, and any possible output $(\hat{\Phi}(X), \hat{y})$,

$$\begin{aligned}&\frac{\Pr[(\mathcal{M}(\Phi(X)) = \hat{\Phi}(X), \mathcal{M}_l(y) = \hat{y})]}{\Pr[(\mathcal{M}(\Phi(X')) = \hat{\Phi}(X), \mathcal{M}_l(y') = \hat{y})]} \\ =&\frac{\Pr[(\mathcal{M}(\Phi(X)) = \hat{\Phi}(X)] \cdot \Pr[\mathcal{M}_l(y) = \hat{y}]}{\Pr[(\mathcal{M}(\Phi(X')) = \hat{\Phi}(X)] \cdot \Pr[\mathcal{M}_l(y') = \hat{y}]} \\ \leq& \exp\left(\epsilon_1 \cdot d_{\measuredangle}(\Phi(X), \Phi(X')) + \epsilon_2\right).\end{aligned}$$

The equality is due to two independent mechanisms $\mathcal{M}$ and $\mathcal{M}_l$. The inequality is from Theorem 1 and the privacy proof of RR. □

B MORE EXPERIMENT RESULTS

Tables 5 and 6 supplement more MIA results; both show significant success rate drops (with different ϵ_1) compared to the non-DP case.

ϵ_1	Entropy		Confidence	
	PurMech	LapMech	PurMech	LapMech
1	0.499	0.502	0.501	0.501
4	0.509	0.508	0.505	0.502
8	0.514	0.513	0.509	0.511
12	0.522	0.524	0.518	0.520
∞	0.634		0.611	

Table 5: Success rates of two MIAs on IMDb

ϵ_1	Entropy		Confidence	
	PurMech	LapMech	PurMech	LapMech
1	0.499	0.502	0.501	0.498
4	0.501	0.499	0.500	0.499
8	0.502	0.501	0.499	0.501
12	0.498	0.502	0.501	0.500
∞	0.630		0.616	

Table 6: Success rates of two MIAs on QNLI