Sentence-level Privacy for Document Embeddings

Anonymous ACL submission

Abstract

User language data can contain highly sensitive personal content. As such, it is imperative to offer users a strong and interpretable privacy guarantee when learning from their data. In this work, we propose SentDP: pure local differential privacy at the sentence level for a single user document. We propose a novel technique, DeepCandidate, that combines concepts from robust statistics and language modeling to produce high-dimensional, general-purpose ϵ-SentDP document embeddings. This guarantees that any single sentence in a document can be substituted with any other sentence while keeping the embedding ϵ-indistinguishable. Our experiments indicate that these private document embeddings are useful for downstream tasks like sentiment analysis and topic classification and even outperform baseline methods with weaker guarantees like word-level Metric DP.

1 Introduction

Language models have now become ubiquitous in NLP (Devlin et al., 2019; Liu et al., 2019b; Alsentzer et al., 2019), pushing the state of the art in a variety of tasks (Strubell et al., 2018; Liu et al., 2019a). While language models capture meaning and various linguistic properties of text (Jawahar et al., 2019; Yenicelik et al., 2020), an individual's written text can include highly sensitive information. Even if such details are not needed or used, sensitive information has been found to be vulnerable and detectable to attacks (Pan et al., 2020; Abdalla et al., 2020; Carlini et al., 2020). Reconstruction attacks (Xie and Hong, 2021) have even successfully broken through private learning schemes that rely on encryption-type methods (Huang et al., 2020).

As of now, there is no broad agreement on what constitutes good privacy for natural language (Kairouz et al., 2019). Huang et al. (2020) argue that different applications and models require different privacy definitions. Several emerging works propose to apply Metric Differential Privacy (Alvim et al., 2018) at the word level (Feyisetan et al., 2019; Feyisetan and Kasiviswanathan, 2021; Carvalho et al., 2021; Qu et al., 2021; Yue et al., 2021; Xu et al., 2021) . They propose to add noise to word embeddings, such that they are indistinguishable from nearby word embeddings.

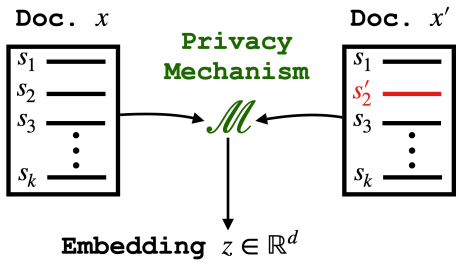


Figure 1: x and x' yield $z \in \mathbb{R}^d$ with similar probability.

At the document level, however, the above definition has two areas for improvement. First, it may not offer the level of privacy desired. Having each word indistinguishable with similar words may not hide higher level concepts in the document, and may not be satisfactory for many users. Second, it may not be very interpretable or easy to communicate to end-users, since the privacy definition relies fundamentally on the choice of embedding model to determine which words are indistinguishable with a given word. This may not be clear and precise enough for end-users to grasp.

In this work, we propose a new privacy definition for documents: sentence privacy. This guarantee is both strong and interpretable: any sentence in a document must be indistinguishable with *any* other sentence. A document embedding is sentence-private if we can replace any single sentence in the document and have a similar probability of producing the same embedding. As such, the embedding only stores limited information unique to any given sentence. This definition is easy to communicate and strictly stronger than word-level definitions, as modifying a sentence can be changing one word.

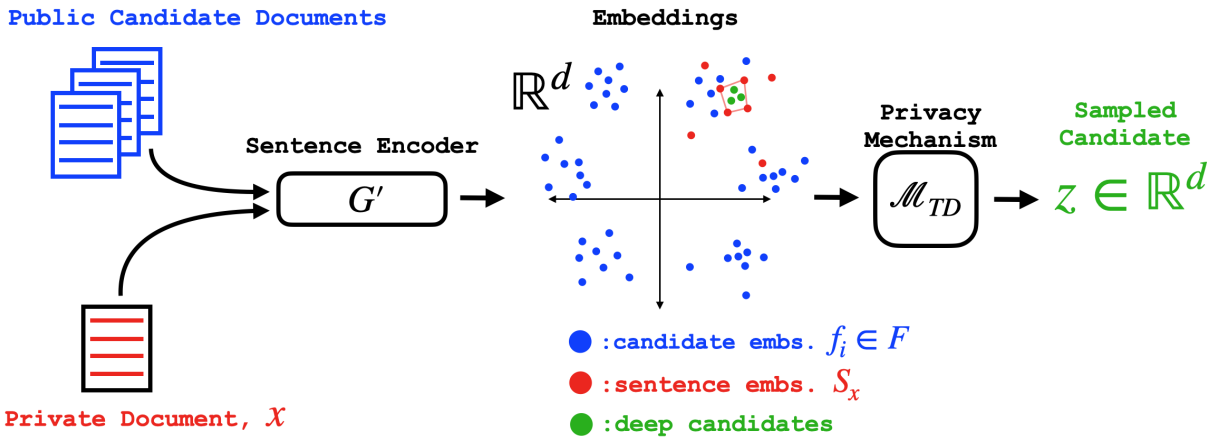


Figure 2: DeepCandidate generates a private embedding z of document x by selecting from a set F of public, non-private document embeddings. Sentences from x are encoded by G'. The privacy mechanism $\mathcal{M}_{\text{TD}}$, then privately samples from F, with a preference for candidates with high Tukey Depth, 'deep candidates'. G' is trained beforehand to ensure that deep candidates are likely to exist and are relevant to x.

Although this definition is strong, we are able to produce unsupervised, general embeddings of documents that are useful for downstream tasks like sentiment analysis and topic classification. To achieve this we propose a novel privacy mechanism, DeepCandidate, which privately samples a high-dimensional embedding from a preselected set of candidate embeddings derived from public, non-private data. DeepCandidate works by first pre-tuning a sentence encoder on public data such that semantically different document embeddings are far apart from each other. Then, we approximate each candidate's Tukey Depth within the private documents' sentence embeddings. Deeper candidates are the most likely to be sampled to represent the private document. We evaluate DeepCandidate on three illustrative datasets, and show that these unsupervised private embeddings are useful for both sentiment analysis and topic classification as compared to baselines.

In summary, this work makes the following contributions to the language privacy literature:

1. A new, strong, and interpretable privacy definition that offers complete indistinguishability to each sentence in a document.
2. A novel, unsupervised embedding technique, DeepCandidate, to generate sentence-private document embeddings.
3. An empirical assessment of DeepCandidate, demonstrating its advantage over baselines, delivering strong privacy and utility.

2 Background and Related Work

Setting. We denote a 'document' as a sequence of sentences. Let $s \in \mathcal{S}$ be any finite-length sentence. Then, the space of all documents is $\mathcal{X} = \mathcal{S}^*$ and document $x \in \mathcal{X}$ is written as $x = (s_1, s_2, \ldots, s_k)$ for any non-negative integer k of sentences. In this work, we focus on cohesive documents of sentences written together like reviews or emails, but our methods and guarantees apply to any sequence of sentences, such as a collection of messages written by an individual over some period of time.

Our task is to produce an embedding $z \in \mathbb{R}^d$ of any document $x \in \mathcal{X}$ such that any single sentence $s_i \in x$ is indistinguishable with every other sentence $s_i' \in \mathcal{S} \backslash s_i$. That is, if one were to replace any single sentence in the document $s_i \in x$ with *any other* sentence $s_i' \in \mathcal{S} \backslash s_i$, the probability of producing a given embedding z is similar. To achieve this, we propose a randomized embedding function (the embedding *mechanism*) $\mathcal{M} : \mathcal{X} \to \mathbb{R}^d$, that generates a private embedding $z = \mathcal{M}(x)$ that is useful for downstream tasks.

2.1 Differential Privacy

The above privacy notion is inspired by Differential Privacy (DP) (Dwork, 2006). It guarantees that — whether an individual participates (dataset D) or not (dataset D') — the probability of any output only chances by a constant factor.

Definition 2.1 (Differential Privacy)**.** Given any pair of datasets $D, D' \in \mathcal{D}$ that differ only in the information of a single individual, we say that the mechanism $\mathcal{A} : \mathcal{D} \to \mathcal{O}$, satisfies ϵ-DP if

$$\Pr[\mathcal{A}(D) \in O] \leq e^{\epsilon} \Pr[\mathcal{A}(D') \in O]$$

for any event $O \subseteq \mathcal{O}$.

Note that we take probability over the randomness of the mechanism $\mathcal{A}$ only, not the data distribution. DP has several nice properties that make it easy to work with including closure under post-processing, an additive privacy budget (composition), and closure under group privacy guarantees

(guarantees to a *subset* of multiple participants). See Dwork et al. 2014 for more details.

When our output space is a discrete and finite set of alternatives to choose from $\mathcal{O} = (o_1, o_2, \ldots, o_n)$, we may use the *exponential mechanism* to satisfy ϵ-DP (McSherry and Talwar, 2007). To do so, we specify a *utility function* over input/output pairs, $u : \mathcal{D} \times \mathcal{O} \to \mathbb{R}$. The utility of choosing alternative $o \in \mathcal{O}$ when the input is dataset $D \in \mathcal{D}$ is then given by $u(D, o)$. The *sensitivity* of $u(\cdot, \cdot)$ is the worst-case change in utility over pairs of neighboring datasets, $\Delta u = \max_{D,D',o} |u(D, o) - u(D', o)|$.

Definition 2.2. The *exponential mechanism* $\mathcal{A}_{\text{Exp}} : \mathcal{D} \to \mathcal{O}$ is a randomized algorithm with output distribution

$$\Pr[\mathcal{A}_{\text{Exp}}(D) = o] \propto \exp\left(\frac{\epsilon u(x, r)}{2\Delta u}\right) \quad .$$

2.2 Related Work

Natural Language Privacy. Previous work has demonstrated that NLP models and embeddings are vulnerable to reconstruction attacks (Carlini et al., 2020; Abdalla et al., 2020; Pan et al., 2020). In response there have been various efforts to design privacy-preserving techniques and definitions across NLP tasks. A line of work focuses on how to make NLP model training satisfy DP (Kerrigan et al., 2020; Bagdasaryan et al., 2019). This is distinct from our work in that it satisfies central DP – where data is first aggregated non-privately and then privacy preserving algorithms (i.e. training) are run on that data. We model this work of the *local* version of DP (Dwork et al., 2006), wherein each individual's data is made private before centralizing. Our definition guarantees privacy to a single document as opposed to a single individual.

A line of work more comparable to our approach makes documents locally private by generating a randomized version of a document that satisfies some formal privacy definition. As with the private embedding of our work, this generates a locally private *representation* of a given document x. The overwhelming majority of these methods satisfy an instance of Metric-DP (Alvim et al., 2018) at the word level (Feyisetan et al., 2019; Feyisetan and Kasiviswanathan, 2021; Carvalho et al., 2021; Qu et al., 2021; Yue et al., 2021; Xu et al., 2021). As discussed in the introduction, this guarantees that a document x is indistinguishable with any other document x' produced by swapping a single word in x with a similar word. Two words are 'similar' if they are close in the word embeddings space (e.g. GloVe). This guarantee is strictly weaker than our proposed definition, SentDP, which offers indistinguishability to any two documents that differ in an entire sentence.

Privacy-preserving embeddings. There is a large body of work on non-NLP privacy-preserving embeddings, as these embeddings have been shown to be vulnerable to attacks (Song and Raghunathan, 2020). Li and Clifton (2021) attempt to generate locally private embeddings by bounding the embedding space, and we compare with this method in our experiments. Kamath et al. (2019) propose a method for privately publishing the average of embeddings, but their algorithm is not suited to operate on the small number of samples (sentences) a given document offers. Finally, Beimel et al. (2019) propose a method for privately learning halfspaces in $\mathbb{R}^d$, which is relevant to private Tukey Medians, but their method would restrict input examples (sentence embeddings) to a finite discrete set in $\mathbb{R}^d$, a restriction we cannot tolerate.

3 Sentence-level Privacy

We now introduce our simple, strong privacy definition, along with concepts we use to satisfy it.

3.1 Definition

In this work, we adopt the *local* notion of DP (Dwork et al., 2006), wherein each individual's data is guaranteed privacy locally before being reported and centralized. Our mechanism $\mathcal{M}$ receives a single document from a single individual, $x \in \mathcal{X}$. We require that $\mathcal{M}$ provides indistinguishability between documents x, x' differing *in one sentence*.

Definition 3.1 (Sentence Privacy, SentDP)**.** Given any pair of documents $x, x' \in \mathcal{X}$ that differ only in one sentence, we say that a mechanism $\mathcal{M} : \mathcal{X} \to \mathcal{O}$ satisfies ϵ-SentDP if

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

for any event $O \subseteq \mathcal{O}$.

We focus on producing an embedding of the given document x, hence the output space is $\mathcal{O} = \mathbb{R}^d$. For instance, consider the neighboring documents $x = (s_1, s_2, \ldots, s_k)$ and $x' = (s_1, s_2', \ldots, s_k)$ that differ in the second sentence, i.e. s_2, s_2' can be *any* pair of sentences in $\mathcal{S}^2$. This is a strong notion of privacy in comparison to existing definitions across NLP tasks. However, we

show that we can guarantee SentDP while still providing embeddings that are useful for downstream tasks like sentiment analysis and classification. In theory, a SentDP private embedding z should be able to encode any information from the document that is not unique to a small subset of sentences. For instance, z can reliably encode the sentiment of x as long as *multiple* sentences reflect the sentiment. By the group privacy property of DP, which SentDP retains, two documents differing in α sentences are $\alpha\epsilon$ indistinguishable. So, if more sentences reflect the sentiment, the more $\mathcal{M}$ can encode this into z without compromising on privacy.

3.2 Sentence Mean Embeddings

Our approach is to produce a private version of the average of general-purpose sentence embeddings. By the post-processing property of DP, this embedding can be used repeatedly in any fashion desired without degrading the privacy guarantee. Our method makes use of existing pre-trained sentence encoding models. We denote this general sentence encoder as $G : \mathcal{S} \to \mathbb{R}^d$. We show in our experiments that the mean of sentence embeddings,

$$\overline{g}(x) = \sum_{s_i \in x} G(s_i) \ , \tag{1}$$

maintains significant information unique to the document and is useful for downstream tasks like classification and sentiment analysis.

We call $\overline{g}(x)$ the *document embedding* since it summarizes the information in document x. While there exist other definitions of document embeddings (Yang et al., 2016; Thongtan and Phienthrakul, 2019; Bianchi et al., 2020), we decide to use averaging as it is a simple and established embedding technique (Bojanowski et al., 2017; Gupta et al., 2019; Li et al., 2020).

3.3 Tukey Depth

Depth is a concept in robust statistics used to describe how central a point is to a distribution. We borrow the definition proposed by Tukey (1975):

Definition 3.2. Given a distribution P over $\mathbb{R}^d$, the Tukey Depth of a point $y \in \mathbb{R}^d$ is

$$\mathrm{TD}_P(y) = \inf_{w \in \mathbb{R}^d} P\{y' : w \cdot (y' - y) \geq 0\} \quad .$$

In other words, take the hyperplane orthogonal to vector w, h_w, that passes through point y. Let P_1^w be the probability under P that a point lands on one side of h_w and let P_2^w be the probability that a point lands on the other side, so $P_1^w + P_2^w = 1$. y is considered deep if $\min(P_1^w, P_2^w)$ is close to a half for *all* vectors w (and thus all h passing through y). The *Tukey Median* of distribution P, $\mathrm{T}_{\mathrm{MED}}(P)$, is the set of all points with maximal Tukey Depth,

$$\mathrm{T}_{\mathrm{MED}}(P) = \arg\max_{y \in \mathbb{R}^d} \mathrm{TD}_P(y) \quad . \tag{2}$$

We only access the distribution P through a finite sample of i.i.d. points, $Y = \{y_1, y_2, \ldots, y_n\}$. The Tukey Depth w.r.t. Y is given by

$$\mathrm{TD}_Y(y) = \inf_{w \in \mathbb{R}^d} |\{y' \in Y : w \cdot (y' - y) \geq 0\}| \ ,$$

and the median, $\mathrm{T}_{\mathrm{MED}}(Y)$, maximizes the depth and is at most half the size of our sample $\left\lfloor \frac{n}{2} \right\rfloor$.

Generally, finding a point in $\mathrm{T}_{\mathrm{MED}}(Y)$ is hard; SOTA algorithms have an exponential dependency in dimension (Chan, 2004), which is a non-starter when working with high-dimensional embeddings. However, there are efficient approximations which we will take advantage of.

4 DeepCandidate

While useful and general, the document embedding $\overline{g}(x)$ does not satisfy SentDP. We now turn to describing our privacy-preserving technique, DeepCandidate, which generates general, ϵ-SentDP document embeddings that preserve relevant information in $\overline{g}(x)$, and are useful for downstream tasks. To understand the nontrivial nature of this problem, we first analyze why the simplest, straightfoward approaches are insufficient.

Motivation. Preserving privacy for high dimensional objects is known to be challenging (Kamath et al., 2019; Feyisetan and Kasiviswanathan, 2021; Zhou et al., 2009) . For instance, adding Laplace noise directly to $\overline{g}(x)$, as done to satisfy some privacy definitions (Feyisetan et al., 2019; Alvim et al., 2018), does not guarantee SentDP for any ϵ. Recall that the embedding space is all of $\mathbb{R}^d$. A change in one sentence can lead to an unbounded change in $\overline{g}(x)$, since we do not put any restrictions on the general encoder G. Thus, no matter how much noise we add to $\overline{g}(x)$ we cannot satisfy SentDP.

A straightforward workaround might be to simply truncate embeddings such that they all lie in a limited set such as a sphere or hypercube as done in prior work (Li and Clifton, 2021; Abadi et al., 2016). In doing so, we bound how far

apart embeddings can be for any two sentences, $\|G(s_i) - G(s_i')\|_1$, thus allowing us to satisfy SentDP by adding finite variance noise. However, such schemes offer poor utility due to the high dimensional nature of useful document embeddings (we confirm this in our experiments). We must add noise with standard deviation proportional to the dimension of the embedding, thus requiring an untenable degree of noise for complex encoders like BERT which embed into $\mathbb{R}^{768}$.

Our method has three pillars: **(1)** sampling from a candidate set of public, non-private document embeddings to represent the private document, **(2)** using the Tukey median to approximate the document embedding, and **(3)** pre-training the sentence encoder, G, to produce relevant candidates with high Tukey depth for private document x.

4.1 Taking advantage of public data: sampling from candidates

Instead of having our mechanism select a private embedding z from the entire space of $\mathbb{R}^d$, we focus the mechanism to select from a set of m candidate embeddings, F, generated by m public, non-private documents. We assume the document x is drawn from some distribution μ over documents $\mathcal{X}$. For example, if we know x is a restaurant review, μ may be the distribution over all restaurant reviews. F is then a collection of document embeddings from m publicly accessible documents $x_i \sim \mu$,

$$F = \{f_i = \overline{g}(x_i) : x_1, \ldots, x_m \overset{\text{iid}}{\sim} \mu\} \ .$$

We denote the corresponding distribution over f_i as $\overline{g}(\mu)$. By selecting documents F to be similar in nature to the private document x, we inject an advantageous inductive bias into our mechanism, which is critical to satisfy strong privacy while preserving meaningful information relevant to x.

4.2 Approximating the document embedding: The Tukey Median

We now propose a novel mechanism $\mathcal{M}_{\text{TD}}$, which approximates $\overline{g}(x)$ by sampling a candidate embedding from F. $\mathcal{M}_{\text{TD}}$ works by concentrating probability on candidates with high Tukey Depth w.r.t. the set of sentence embeddings $S_x = \{G(s_i) : s_i \in x\}$. We model sentences s_i from document x as i.i.d. draws from distribution ν_x. Then, S_x is k draws from $g(\nu_x)$, the distribution of sentences from ν_x passing through G. Deep points are a good approximation of the mean under light assumptions. If $g(\nu_x)$ belongs to the set of halfspace-symmetric distributions (including all elliptic distributions e.g. Gaussians), we know that its mean lies in the Tukey Median (Zhu et al., 2020).

Formally, $\mathcal{M}_{\text{TD}}$ is an instance of the exponential mechanism (Definition 2.2), and is defined by its utility function. We set the utility of a candidate document embedding $f_i \in F$ to be an approximation of its depth w.r.t. sentence embeddings S_x,

$$u(x, f_i) = \widehat{\text{TD}}_{S_x}(f_i) \quad . \tag{3}$$

The approximation $\widehat{\text{TD}}_{S_x}$, which we detail in the Appendix, is necessary for computational efficiency. If the utility of f_i is high, we call it a 'deep candidate' for sentence embeddings S_x.

The more candidates sampled (higher m), the higher the probability that at least one has high depth. Without privacy, we could report the deepest candidate, $z = \arg\max_{f_i \in F} \widehat{\text{TD}}_{S_x}(f_i)$. However, when preserving privacy with $\mathcal{M}_{\text{TD}}$, increasing m has diminishing returns. To see this, fix a set of sentence embeddings S_x for document x and the i.i.d. distribution over candidate embeddings $f_i \sim \overline{g}(\mu)$. This induces a multinomial distribution over depth,

$$u_j = \Pr[u(x, f_i) = j], \ \sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} u_j = 1 \ ,$$

where randomness is taken over draws of f_i.

For candidate set F and sentence embeddings S_x, the probability of $\mathcal{M}_{\text{TD}}$'s selected candidate, z, having (approximated) depth j^* is given by

$$\Pr[u(x, z) = j^*] = \frac{a_{j^*} e^{\epsilon j^*/2}}{\sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} a_j e^{\epsilon j/2}} \tag{4}$$

where a_j is the fraction of candidates in F with depth j. For m sufficiently large, a_j concentrates around u_j. Thus, while increasing m may increase the likelihood of a deep candidate existing, it does not increase the probability of *sampling* a deep candidate.

Table 1: Conditions for deep candidates

ϵ	b	j^*
3	55	5
6	25	3
10	5	2
23	1	1

For numerical intuition, suppose $m = 5000$ (as in our experiments), $\geq b$ candidates have depth

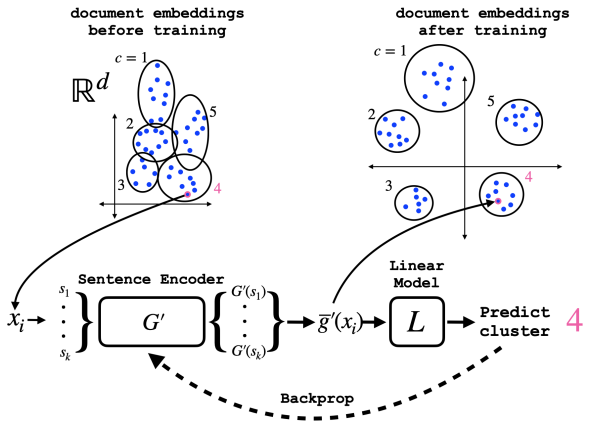


Figure 3: G' is trained to encourage similar documents to embed close together and different documents to embed far apart. We first compute embeddings of all (public, non-private) training set documents T with pretrained encoder G, $T_G = \{t_i = \overline{g}(x_i) : x_i \in T\}$ (blue dots). We run k-means to define n_c clusters, and label each training document embedding $t_i \in T_G$ with its cluster c. We then train H to recode sentences to S'_x such that their mean $\overline{g}'(x)$ can be used by a linear model L to predict cluster c. Our training objective is the cross-entropy loss of the linear model L in predicting c.

$\geq j^*$, and all other candidates have depth 0, $\mathcal{M}_{\text{TD}}$ will sample one of these deep candidates w.p. $\geq$ 0.95 under the settings in Table 1.

For low $\epsilon < 10$ (high privacy), about 1% of candidates need to have high depth (≥ 3) in order to be reliably sampled. Note that this is only possible for documents with ≥ 6 sentences. For higher $\epsilon \geq 10$, $\mathcal{M}_{\text{TD}}$ will reliably sample low depth candidates even if there are only a few.

From these remarks we draw two insights on how DeepCandidate can achieve high utility.
(1) *More sentences* A higher k enables greater depth, and thus a higher probability of sampling deep candidates with privacy. We explore this effect in our experiments.
(2) *Tuned encoder* By tuning the sentence encoder G for a given domain, we can modify the distribution over document embeddings $\overline{g}(\mu)$ and sentence embeddings $g(\nu_x)$ to encourage deep candidates (high probability u_j for deep j) that are relevant to document x.

4.3 Taking advantage of structure: cluster-preserving embeddings

So far, we have identified that deep candidates from F can approximate $\overline{g}(x)$. To produce a good approximation, we need to ensure that 1) there reliably exist deep candidates for any given set of sentence embeddings S_x, and 2) that these deep candidates are good representatives of document x. The general sentence encoder G used may not satisfy this 'out of the box'. If the distribution on document embeddings $\overline{g}(\mu)$ is very scattered around the instance space $\mathbb{R}^{768}$, it can be exceedingly unlikely to have a deep candidate f_i among sentence embeddings S_x. On the other hand, if distribution $\overline{g}(\mu)$ is tightly concentrated in one region (e.g. 'before training' in Figure 3), then we may reliably have many deep candidates, but several will be poor representatives of the document embedding $\overline{g}(x)$.

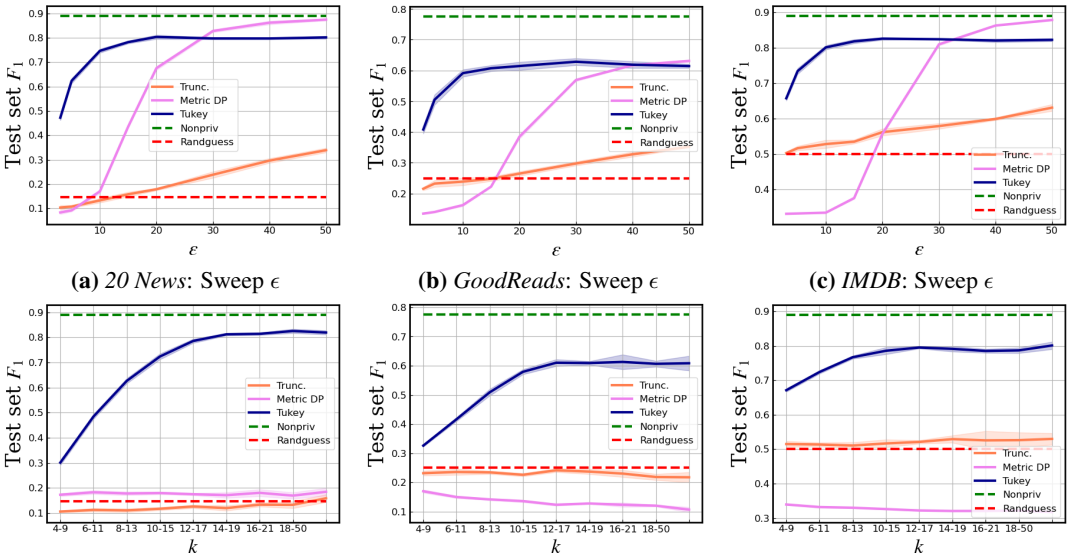
To prevent this, we propose an unsupervised, efficient, and intuitive modification to the (pretrained) sentence encoder G. We freeze the weights of G and add additional perceptron layers mapping into the same embeddings space $H : \mathbb{R}^d \to \mathbb{R}^d$, producing the extended encoder $G' = H \circ G$. Broadly, we train H to place similar document embeddings close together, and different embeddings far part. To do so, we leverage the assumption that a given domain's distribution over document embeddings $\overline{g}(\mu)$ can be parameterized by n_c clusters, visualized as the black circles in Figure 3. H's aim is to recode sentence embeddings such that document embedding clusters are preserved, but spaced apart from each other. By preserving clusters, we are more likely to have deep candidates (increased probability u_j for high depth j). By spacing clusters apart, these deep candidates are more likely to come from the same or a nearby cluster as document x, and thus be good representatives. Note that H is domain-specific: we train separate H encoders for each dataset.

4.4 Sampling Algorithm

The final component of DeepCandidate is computing the approximate depth of a candidate for use as utility in the exponential mechanism as in Eq. (3). We use a version of the approximation algorithm proposed in Gilad-Bachrach and Burges 2012. Intuitively, our algorithm computes the one-dimensional depth of each f_i among x's sentence embeddings S_x on each of p random projections. The approximate depth of f_i is then its lowest depth across the p projections. We are guaranteed that $\widehat{\text{TD}}_{S_x}(f_i) \geq \text{TD}_{S_x}(f_i)$. Due to space constraints, we leave the detailed description of the algorithm for the Appendix.

Theorem 4.1. *$\mathcal{M}_{TD}$ satisfies ϵ-Sentence Privacy*

Proof follows from the fact that $\widehat{\text{TD}}_{S_x}(f_i)$ has bounded sensitivity (changing one sentence can only change depth of f_i by one). We expand on this, too, in the Appendix.



(a) *20 News*: Sweep ϵ **(b)** *GoodReads*: Sweep ϵ **(c)** *IMDB*: Sweep ϵ

(d) *20 News*: Sweep k **(e)** *GoodReads*: Sweep k **(f)** *IMDB*: Sweep k

Figure 4: Comparison of our mechanism with two baselines: truncation (Li and Clifton, 2021) and word-level Metric DP (Feyisetan et al., 2019) for both sentiment analysis (*IMDB*) and topic classification (*GoodReads*, *20News*) on private, unsupervised embeddings. All plots show test-set macro F_1 scores. The top row shows performance vs. privacy parameter ϵ (lower is better privacy). The bottom row shows performance vs. number of sentences k with $\epsilon = 10$. DeepCandidate outperforms both baselines across datasets and tasks. Note that at a given ϵ, word-level Metric-DP is a significantly weaker privacy guarantee.

5 Experiments

5.1 Datasets

We produce private, general embeddings of documents from three English-language datasets:

Good Reads (Wan and McAuley, 2018) 60k book reviews from four categories: fantasy, history, romance, and childrens literature. Train-48k | Val-8k | Test-4k

20 News Groups (Lang, 1995) 11239 correspondences from 20 different affinity groups. Due to similarity between several groups (e.g. `comp.os.ms-windows.misc` and `comp.sys.ibm.pc.hardware`), the dataset is partitioned into nine categories. Train-6743k | Val-2247k | Test-2249k

IMDB (Maas et al., 2011) 29k movie reviews from the IMDB database, each labeled as a positive or negative review. Train-23k | Val-2k | Test-4k

To evaluate utility of these unsupervised, private embeddings, we check if they are predictive of document properties. For the *Good Reads* and *20 News Groups* datasets, we evaluate how useful the embeddings are for topic classification. For *IMDB* we evaluate how useful the embeddings are for sentiment analysis (positive or negative review). Our metric for performance is test-set macro F_1 score.

5.2 Training Details & Setup

For the general encoder, $G : \mathcal{S} \to \mathbb{R}^{768}$, we use SBERT (Reimers and Gurevych, 2019), a version of BERT fine-tuned for sentence encoding. Sentence embeddings are generated by mean-pooling output tokens. In all tasks, we freeze the weights of SBERT. The cluster-preserving recoder, H, as well as every classifier is implemented as an instance of a 4-layer MLP taking 768-dimension inputs and only differing on output dimension. We denote an instance of this MLP with output dimension o as $\mathbf{MLP}^o$. We run 5 trials of each experiment with randomness taken over the privacy mechanisms, and plot the mean along with a $\pm$ 1 standard deviation envelope.

DeepCandidate: The candidate set F consists of 5k document embeddings from the training set, each containing at least 8 sentences. To train G', we find $n_c = 50$ clusters with k-means. We train a classifier $C_{\text{dc}} = \mathbf{MLP}^r$ on document embeddings $g'(x)$ to predict class, where r is the number of classes (topics or sentiments).

5.3 Baselines

We compare the performance of DeepCandidate with 4 baselines: **Non-private**, **Truncation**, **Word-level Metric-DP**, and **Random Guesser**.

Non-private: This demonstrates the usefulness of non-private sentence-mean document embeddings $\overline{g}(x)$. We generate $\overline{g}(x)$ for every document using SBERT, and then train a classifier $C_{\text{nonpriv}} = \mathbf{MLP}^r$ to predict x's label from $\overline{g}(x)$.

Truncation: We adopt the method from Li and

Clifton 2021 to truncate (clip) sentence embeddings within a box in $\mathbb{R}^{768}$, thereby bounding sensitivity as described at the beginning of Section 4. Laplace noise is then added to each dimension. Documents with more sentences have proportionally less noise added due to the averaging operation reducing sensitivity.

Word Metric-DP (MDP): The method from Feyisetan et al. 2019 satisfies ϵ-word-level metric DP by randomizing words. We implement MDP to produce a randomized document x', compute $\overline{g}(x')$ with SBERT, and predict class using C_{nonpriv}.

Random Guess: To set a bottom-line, we show the theoretical performance of a random guesser only knowing the distribution of labels.

5.4 Results & Discussion

How does performance change with privacy parameter ϵ?

This is addressed in Figures 4a to 4c. Here, we observe how the test set macro F_1 score changes with privacy parameter ϵ (a lower ϵ offers stronger privacy). Generally speaking, for local differential privacy, $\epsilon < 10$ is taken to be a strong privacy regime, $10 \leq \epsilon < 20$ is moderate privacy, and $\epsilon \geq 25$ is weak privacy. The **truncation** baseline mechanism does increase accuracy with increasing ϵ, but never performs much better than the random guesser. This is to be expected with high dimension embeddings, since the standard deviation of noise added increases linearly with dimension.

The word-level **MDP** mechanism performs significantly better than **truncation**, achieving relatively good performance for $\epsilon \geq 30$. There are two significant caveats, however. First, is the privacy definition: as discussed in the Introduction, for the same ϵ, word-level MDP is strictly weaker than SentDP. The second caveat is the level of ϵ at which privacy is achieved. Despite a weaker privacy definition, the MDP mechanism does not achieve competitive performance until the weak-privacy regime of ϵ. We suspect this is due to two reasons. First, is the fact that the MDP mechanism does not take advantage of contextual information in each sentence as our technique does; randomizing each word independently does not use higher level linguistic information. Second, is the fact that the MDP mechanism does not use domain-specific knowledge as our mechanism does with use of relevant candidates and domain specific sentence encodings.

In comparison, DeepCandidate offers strong utility across tasks and datasets for relatively low values of ϵ, even into the strong privacy regime. Beyond $\epsilon = 25$, the performance of DeepCandidate tends to max out, approximately 10-15% below the non-private approach. This is due to the fact that DeepCandidate offers a noisy version of an *approximation* of the document embedding $\overline{g}(x)$ – it cannot perform any better than deterministically selecting the deepest candidate, and even this candidate may be a poor representative of x. We consider this room for improvement, since there are potentially many other ways to tune G' and select the candidate pool F such that deep candidates are nearly always good representatives of a given document x.

How does performance change with the number of sentences k?

This is addressed in Figures 4d to 4f. We limit the test set to those documents with k in the listed range on the x-axis. We set $\epsilon = 10$, the limit of the strong privacy regime. Neither baseline offers performance above that of the random guesser at this value of ϵ. DeepCandidate produces precisely the performance we expect to see: documents with more sentences result in sampling higher quality candidates, confirming the insights of Section 4.2. Across datasets and tasks, documents with more than 10-15 sentences tend to have high quality embeddings.

6 Conclusions and Future Work

We introduce a strong and interpretable local privacy guarantee for documents, SentDP, along with DeepCandidate, a technique that combines principles from NLP and robust statistics to generate general ϵ-SentDP embeddings. Our experiments confirm that such methods can outperform existing approaches even with with more relaxed privacy guarantees. Previous methods have argued that it is "virtually impossible" to satisfy pure local DP (Feyisetan et al., 2019; Feyisetan and Kasiviswanathan, 2021) at the word level while capturing linguistic semantics. Our work appears to refute this notion at least for documents.

To follow up, we plan to explore other approaches (apart from k-means) of capturing the structure of the embedding distribution $\overline{g}(\mu)$ to encourage better candidate selection. We also plan to experiment with decoding private embeddings back to documents by using novel candidates produced by a generative model trained on F.

References

Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep Learning with Differential Privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318. ArXiv: 1607.00133.

Mohamed Abdalla, Moustafa Abdalla, Graeme Hirst, and Frank Rudzicz. 2020. Exploring the Privacy-Preserving Properties of Word Embeddings: Algorithmic Validation Study. *Journal of Medical Internet Research*, 22(7):e18055.

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical bert embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78.

Mário Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. 2018. Invited Paper: Local Differential Privacy on Metric Spaces: Optimizing the Trade-Off with Utility. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 262–267. ISSN: 2374-8303.

Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. 2019. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, 32:15479–15488.

Amos Beimel, Shay Moran, Kobbi Nissim, and Uri Stemmer. 2019. Private Center Points and Learning of Halfspaces. *arXiv:1902.10731 [cs, stat]*. ArXiv: 1902.10731.

Federico Bianchi, Silvia Terragni, and Dirk Hovy. 2020. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2020. Extracting Training Data from Large Language Models. *arXiv:2012.07805 [cs]*. ArXiv: 2012.07805.

Ricardo Silva Carvalho, Theodore Vasiloudis, and Oluwaseyi Feyisetan. 2021. Tem: High utility metric differential privacy on text. *arXiv preprint arXiv:2107.07928*.

Timothy M Chan. 2004. An optimal randomized algorithm for maximum tukey depth. In *SODA*, volume 4, pages 430–436.

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

Cynthia Dwork. 2006. *Differential Privacy*, volume 4052.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. 2006. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004, pages 486–503. Springer Berlin Heidelberg, Berlin, Heidelberg. Series Title: Lecture Notes in Computer Science.

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

Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. 2019. Privacy- and Utility-Preserving Textual Analysis via Calibrated Multivariate Perturbations.

Oluwaseyi Feyisetan and Shiva Kasiviswanathan. 2021. Private release of text embedding vectors. In *Proceedings of the First Workshop on Trustworthy Natural Language Processing*, pages 15–27.

Ran Gilad-Bachrach and Chris J. C. Burges. 2012. The Median Hypothesis.

Prakhar Gupta, Matteo Pagliardini, and Martin Jaggi. 2019. Better word embeddings by disentangling contextual n-gram information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 933–939.

Yangsibo Huang, Zhao Song, Danqi Chen, Kai Li, and Sanjeev Arora. 2020. TextHide: Tackling data privacy in language understanding tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1368–1382, Online. Association for Computational Linguistics.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*.

Gautam Kamath, Jerry Li, Vikrant Singhal, and Jonathan Ullman. 2019. Privately Learning High-Dimensional Distributions. In *Conference on Learning Theory*, pages 1853–1902. PMLR. ISSN: 2640-3498.

Gavin Kerrigan, Dylan Slack, and Jens Tuyls. 2020. Differentially Private Language Models Benefit from Public Pre-training. *arXiv:2009.05886 [cs]*. ArXiv: 2009.05886.

Ken Lang. 1995. Home Page for 20 Newsgroups Data Set.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from bert for semantic textual similarity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9119–9130.

Tao Li and Chris Clifton. 2021. Differentially Private Imaging via Latent Space Manipulation. *arXiv:2103.05472 [cs]*. ArXiv: 2103.05472.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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 *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Frank McSherry and Kunal Talwar. 2007. Mechanism Design via Differential Privacy.

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

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. 2021. Privacy-Adaptive BERT for Natural Language Understanding. *arXiv:2104.07504 [cs]*. ArXiv: 2104.07504.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs]*. ArXiv: 1908.10084.

Congzheng Song and Ananth Raghunathan. 2020. Information Leakage in Embedding Models. *arXiv:2004.00053 [cs, stat]*. ArXiv: 2004.00053.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038.

Tan Thongtan and Tanasanee Phienthrakul. 2019. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 407–414.

John W Tukey. 1975. Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematicians, Vancouver, 1975*, volume 2, pages 523–531.

Mengting Wan and Julian J. McAuley. 2018. Item recommendation on monotonic behavior chains. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 86–94. ACM.

Shangyu Xie and Yuan Hong. 2021. Reconstruction attack on instance encoding for language understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2038–2044.

Nan Xu, Oluwaseyi Feyisetan, Abhinav Aggarwal, Zekun Xu, and Nathanael Teissier. 2021. Density-Aware Differentially Private Textual Perturbations Using Truncated Gumbel Noise. *The International FLAIRS Conference Proceedings*, 34(1).

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489.

David Yenicelik, Florian Schmidt, and Yannic Kilcher. 2020. How does bert capture semantics? a closer look at polysemous words. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 156–162.

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. *arXiv:2106.01221 [cs]*. ArXiv: 2106.01221.

Shuheng Zhou, Katrina Ligett, and Larry Wasserman. 2009. Differential privacy with compression. In *2009 IEEE International Symposium on Information Theory*, pages 2718–2722. ISSN: 2157-8117.

Banghua Zhu, Jiantao Jiao, and Jacob Steinhardt. 2020. When does the Tukey Median work? In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 1201–1206, Los Angeles, CA, USA. IEEE.

A Appendix

A.1 Privacy Mechanism

We now describe in detail our instance of the exponential mechanism $\mathcal{M}_{\text{TD}}$. Recall from Definition 2.2 that the exponential mechanism samples candidate $f_i \in F$ with probability

$$\Pr[\mathcal{M}(x) = f_i] \propto \exp\left(\frac{\epsilon u(x, f_i)}{2\Delta u}\right) .$$

Thus, $\mathcal{M}_{\text{TD}}$ is fully defined by its utility function, which, as listed in Equation (3), is approximate Tukey Depth,

$$u(x, f_i) = \widehat{\text{TD}}_{S_x}(f_i) \quad .$$

Our approximation of Tukey Depth $\widehat{\text{TD}}_{S_x}(f_i)$ is described in Algorithm 1 which is an adaptation of the general median hypothesis algorithm proposed by Gilad-Bachrach and Burges (2012). As an abuse of notation, we write s_i to be the sentence embedding written as $G(s_i)$ in the main paper.

Note that we can precompute the projections on line 10. The runtime is $O(mkp)$: for each of m candidates and on each of p projections, we need to compute the scalar difference with k sentence embeddings. Sampling from the multinomial distribution defined by P_F then takes $O(m)$ time.

Additionally note from lines 13 and 15 that utility has a maximum of 0 and a minimum of $-\frac{k}{2}$, which is a semantic change from the main paper where maximum utility is $\frac{k}{2}$ and minimum is 0.

A.2 Proof of Privacy

Theorem 4.1 *$\mathcal{M}_{TD}$ satisfies ϵ-Sentence Privacy*

Proof. It is sufficient to show that the sensitivity,

$$\Delta u = \max_{x,x',f_i} |u(x, f_i) - u(x', f_i)| \leq 1 \quad .$$

Let us expand the above expression using the terms in Algorithm 1.

$$\begin{aligned}
\Delta u &= \max_{x,x',f_i} |\max_{j\in[p]} u_j(x, f_i) - \max_{j'\in[p]} u_{j'}(x', f_i)| \\
&= \max_{x,x',f_i} |\max_{j\in[p]} \left|h_j(x, f_i) - \frac{k}{2}\right| \\
&- \max_{j'\in[p]} \left|h_{j'}(x', f_i) - \frac{k}{2}\right||\\
&\leq \max_{f_i} |\max_{j\in[p]} \left|h_j(x, f_i) - \frac{k}{2}\right| \\
&- \left(\max_{j'\in[p]} \left|h_{j'}(x, f_i) - \frac{k}{2}\right| + 1\right)| \\
&\leq 1
\end{aligned}$$

Algorithm 1: $\mathcal{M}_{\text{TD}}$ compute probabilities

Input : m candidates F,
sentence embs. $S_x = (s_1, \ldots, s_k)$,
number of projections p

Output : probability of sampling each candidate $P_F = [P_{f_1}, \ldots, P_{f_m}]$

1 $v_1, \ldots, v_p \leftarrow$ random vecs. on unit sphere
2 `// Project all embeddings`
3 **for** $i \in [k]$ **do**
4 **for** $j \in [p]$ **do**
5 $s_i^j \leftarrow s_i^\intercal v_j$
6 **end for**
7 **end for**
8 **for** $i \in [m]$ **do**
9 **for** $j \in [p]$ **do**
10 $f_i^j \leftarrow f_i^\intercal v_j$
11 `/* Compute depth of` f_i `on projection` v_j `*/`
12 $h_j(x, f_i) \leftarrow \#\{s_l^j : s_l^j \geq f_i^j, l \in [k]\}$
13 $u_j(x, f_i) \leftarrow \left|h_j(x, f_i) - \frac{k}{2}\right|$
14 **end for**
15 $u(x, f_i) \leftarrow -\max_{j\in[p]} u_j(x, f_i)$
 $\hat{P}_{f_i} \leftarrow \exp(\epsilon u(x, f_i)/2)$
16 **end for**
17 $\Psi \leftarrow \sum_{i=1}^{m} \hat{P}_{f_i}$
18 **for** $i \in [m]$ **do**
19 $P_{f_i} \leftarrow \frac{1}{\Psi}\hat{P}_{f_i}$
20 **end for**
21 **return** P_F

The last step follows from the fact that $|h_j(x, f_i) - h_j(x', f_i)| \leq 1$ for all $j \in [p]$. In other words, by modifying a single sentence embedding, we can only change the number of embeddings greater than f_i^j on projection j by 1. So, the distance of $h_j(x, f_i)$ from $\frac{k}{2}$ can only change by 1 on each projection. In the 'worst case', the distance $\left|h_j(x, f_i) - \frac{k}{2}\right|$ increases (or reduces) by 1 on every projection v_j. Even then, the maximum distance from $\frac{k}{2}$ across projections (the worst case depth) can only change by 1, giving us a sensitivity of 1. □

A.3 Experimental Details

Here, we provide an extended, detailed version of section 5.

For the general encoder, $G : \mathcal{S} \to \mathbb{R}^{768}$, we use SBERT (Reimers and Gurevych, 2019), a version of BERT fine-tuned for sentence encoding. Sentence embeddings are generated by mean-pooling output tokens. In all tasks, we freeze the weights of SBERT. The cluster-preserving recoder, H, as well as every classifier is implemented as an instance of a 4-layer MLP taking 768-dimension inputs and only differing on output dimension. We denote an instance of this MLP with output dimension o as $\mathbf{MLP}^o$. We run 5 trials of each experiment with randomness taken over the privacy mechanisms, and plot the mean along with a $\pm$ 1 standard deviation envelope.

Non-private: For our non-private baseline, we demonstrate the usefulness of sentence-mean document embeddings. First, we generate the document embeddings $\overline{g}(x_i)$ for each training, validation, and test set document using SBERT, G. We then train a classifier $C_{\text{nonpriv}} = \mathbf{MLP}^r$ to predict each document's topic or sentiment, where r is the number of classes. The number of training epochs is determined with the validation set.

DeepCandidate: We first collect the candidate set F by sampling 5k document embeddings from the subset of the training set containing at least 8 sentences. We run k-means with $n_c = 50$ cluster centers, and label each training set document embedding $t_i \in T_G$ with its cluster. The sentence recoder, $H = \mathbf{MLP}^{768}$ is trained on the training set along with the linear model L with the Adam optimizer and cross-entropy loss. For a given document x, its sentence embeddings S_x are passed through H, averaged together, and then passed to L to predict x's cluster. L's loss is then back-propagated through H. A classifier $C_{\text{dc}} = \mathbf{MLP}^r$ is trained in parallel using a separate instance of the Adam optimizer to predict class from the recoded embeddings, where r is the number of classes (topics or sentiments). The number of training epochs is determined using the validation set. At test time, (generating private embeddings using $\mathcal{M}_{\text{TD}}$), the optimal number of projections p is empirically chosen for each ϵ using the validation set.

Truncation: The truncation baseline (Li and Clifton, 2021) requires first constraining the embedding instance space. We do so by computing the 75% median interval on each of the 768 dimensions of training document embeddings T_G. Sentence embeddings are truncated at each dimension to lie in this box. In order to account for this distribution shift, a new classifier $C_{\text{trunc}} = \mathbf{MLP}^r$ is trained on truncated mean embeddings to predict class. The number of epochs is determined with the validation set. At test time, a document's sentence embeddings S_x are truncated and averaged. We then add Laplace noise to each dimension with scale factor $\frac{768w}{k\epsilon}$, where w is the width of the box on that dimension (*sensitivity* in DP terms). Note that the standard deviation of noise added is inversely proportional to the number of sentences in the document, due to the averaging operation reducing sensitivity.

Word Metric-DP: Our next baseline satisfies ϵ-word-level metric DP and is adopted from (Feyisetan et al., 2019). The corresponding mechanism $\text{MDP} : \mathcal{X} \to \mathcal{X}$ takes as input a document x and returns a private version, x', by randomizing each word individually. For comparison, we generate document embeddings by first randomizing the document $x' = \text{MDP}(x)$ as prescribed by (Feyisetan et al., 2019), and then computing its document embedding $\overline{g}(x')$ using SBERT. At test time, we classify the word-private document embedding using C_{nonpriv}.

Random Guess: To set a bottom-line, we show the theoretical performance of a random guesser. The guesser chooses class i with probability q_i equal to the fraction of i labels in the training set. The performance is then given by $\sum_{i=1}^{r} q_i^2$.

A.4 Reproducability Details

We plan to publish a repo of code used to generate the exact figures in this paper (random seeds have been set) with the final version. Since we do

not train the BERT base model G, our algorithms and training require relatively little computational resouces. Our system includes a single Nvidia GeForce RTX 2080 GPU and a single Intel i9 core. All of our models complete an epoch training on all datasets in less than one minute. We never do more than 20 epochs of training. All of our classifier models train (including linear model) have less than 11 million parameters. The relatively low amount of parameters is due to the fact that we freeze the underlying language model. The primary hyperparameter tuned is the number of projections p. We take the argmax value on the validation set between 10 and 100 projections. We repeat this for each value of ϵ.

Dataset preprocessing: For all datasets, we limit ourselves to documents with at least 2 sentences.

IMDB: This dataset has pre-defined train/test splits. We use the entire training set and form the test set by randomly sampling 4,000 from the test set provided. We do this for efficiency in computing the Metric-DP baseline, which is the slowest of all algorithms performed. Since the Metric-DP baseline randomizes first, we cannot precompute the sentence embeddings $G(s_i)$ – we need to compute the sentence embeddings every single time we randomize. Since we randomize for each sentence of each document at each ϵ and each k over 5 trials – this takes a considerable amount of time.

Good Reads: This dataset as provided is quite large. We randomly sample 15000 documents from each of 4 classes, and split them into 12K training examples, 2K validation examples, and 1K test examples per class.

20 News Groups: We preprocess this dataset to remove all header information, which may more directly tell information about document class, and only provide the model with the sentences from the main body. We use the entire dataset, and form the Train/Val/Test splits by random sampling.