Differential Privacy
and Privacy-Preserving Learning for NLP

Faiaz Rahman
CS 677: Advanced Natural Language Processing
Dr. Dragomir Radev
21 September 2021

Background

Understanding Differential Privacy

Intuition of Differential Privacy

- Non-technical definition: “Differential privacy is a system for publicly sharing information about a dataset by **describing the patterns of groups** within the dataset **while withholding information about individuals** in the dataset”
    - This means that the *presence of one person’s data* in the database will not affect the probability of a result when compared to a database without that person’s data

Differential privacy
Wikipedia [site]

Intuition of Differential Privacy

- Originally defined by **Cynthia Dwork** (Microsoft Research)

> A statistic is a quantity computed from a sample. If a database is a representative sample of an underlying population, the goal of a privacy-preserving statistical database is to enable the user to learn properties of the population as a whole, while protecting the privacy of the individuals in the sample. The work discussed herein was originally motivated by exactly this problem: how to reveal useful information about the underlying population, as represented by the database, while preserving the privacy of individuals.

Differential Privacy
Cynthia Dwork [paper]

Intuition of Differential Privacy

- Apple's overview of how differential privacy works

> The differential privacy technology used by Apple is rooted in the idea that statistical noise that is slightly biased can mask a user's individual data before it is shared with Apple. If many people are submitting the same data, the noise that has been added can average out over large numbers of data points, and Apple can see meaningful information emerge.

Differential Privacy Technical Overview
Apple [article]

Simple Example of a Privacy Attack

In order to avoid such data leaks, we add a controlled amount of statistical noise to obscure the data contributions from individuals in the data set.

Meaning, the donors are asked to add any value between say -100 to 100 to their original age before submitting it or before encrypting their age. Say John Doe added -30 to his original age i.e., 21, the age registered before encryption would be -9.

This might sound crazy?! But, interestingly, by the **Law of Large Numbers** in probability and statistics, it is seen that when the average of these statistically collected data is taken, the noise cancels out and the average obtained is near to the **true average** (average of the data without adding noise (random number))

Now, even if Adam were to reverse engineer John Doe's age, -9 would not make any sense, thus, preserving John Doe's Privacy at the same time allowing Adam to find the average age of the donor.

Simple Example of a Privacy Attack

- things to keep in mind
  - differential privacy is *not* a property of databases, but a property of queries!
    - it helps provide **output privacy** (i.e. how much insight someone can gain on the input by reverse-engineering from the output)
  - the level of privacy is dependent on the **privacy budget** (quantified by the privacy parameter, which we'll see in a bit)
  - in practice, the noise is taken from the Laplace distribution

Data Security vs. Data Privacy

Data Security	Data Privacy
confidentiality data being stored is safe from unauthorized access and use	**traceability** can you verify the history, location, or use of the data?
reliability data is reliable and accurate	**link-ability** can you link all the events or records that belong to the same data subject together?
availability data is available for use when it is needed	**identifiability** can the data be used to find the personal identification of individuals?

Using Differential Privacy to Build Secure Models – Tools, Methods, Best practices
Shaistha Fathima (2021) [Neptune.ai blog]

Simple Example of a Privacy Attack

Adam would like to find the average age of the donor, donating to his XYZ Organisation. At this point, this might seem OK! But, this data can *also potentially be used to find the age of one specific donor*. Say for simplicity, 1000 people made a donation, for which the average age of the donor was found to be 28 years.

Now, just by excluding John Doe's data from the database for the same query, let us assume that the average has changed to 28.007 for 999 donors. From this, Adam could easily find that John Doe is 21 years old. (1000*28 – 999*28.007 = 21.007) Similarly, Adam can repeat the process for other donors to find their actual ages.

Using Differential Privacy to Build Secure Models – Tools, Methods, Best practices
Shaistha Fathima (2021) [Neptune.ai blog]

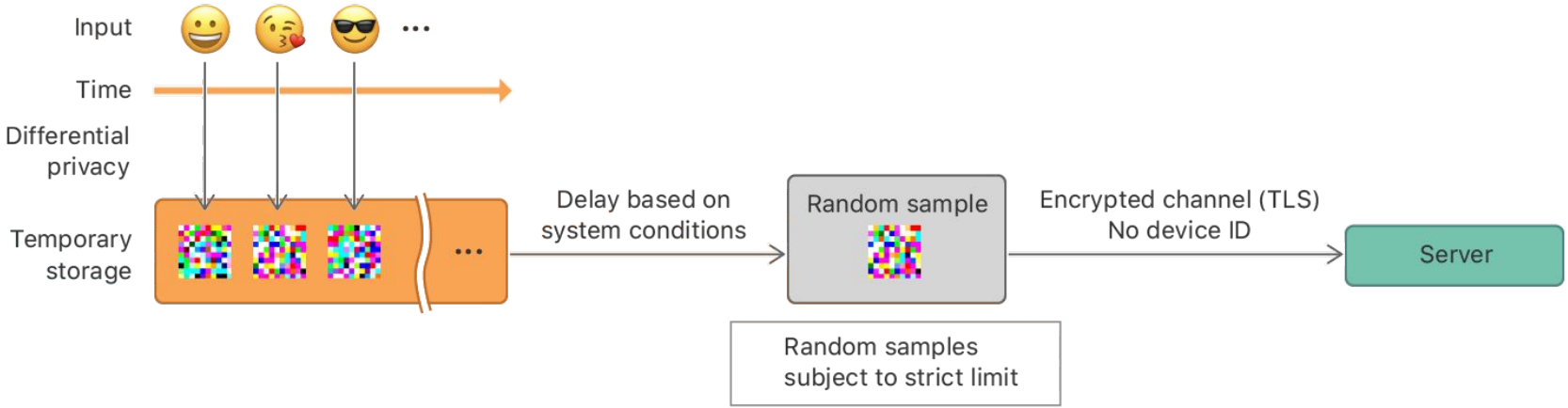
Applications of Differential Privacy

- **US Census:** used to ensure privacy on block-level residential population data in their OnTheMap application
  - *Protecting the Confidentiality of America's Statistics: Adopting Modern Disclosure Avoidance Methods at the Census Bureau* [article]

- **Microsoft:** used to mask the location of individuals in their geolocation databases using a data manipulation technique they call PrivTree
  - *Project PrivTree: Blurring your "where" for location privacy* [article]

Applications of Differential Privacy

- **Apple:** used when discovering popular emojis, identifying resource-intensive websites accessed via Safari, and discovering the use of new words
  - *Learning with Privacy at Scale* [article]



Applications of Differential Privacy

- **Apple:** differential privacy overview

Apple uses local differential privacy to help protect the privacy of user activity in a given time period, while still gaining insight that improves the intelligence and usability of such features as:

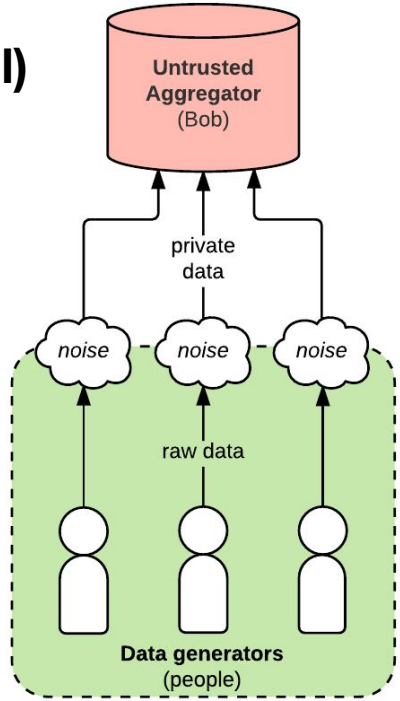
- QuickType suggestions
- Emoji suggestions
- Lookup Hints
- Safari Energy Draining Domains
- Safari Autoplay Intent Detection (macOS High Sierra)
- Safari Crashing Domains (iOS 11)
- Health Type Usage (iOS 10.2)

“Differential privacy” describes a promise, made by a data holder, or curator, to a data subject: “You will not be affected, adversely or otherwise, by allowing your data to be used in any study or analysis, no matter what other studies, data sets, or information sources, are available.”

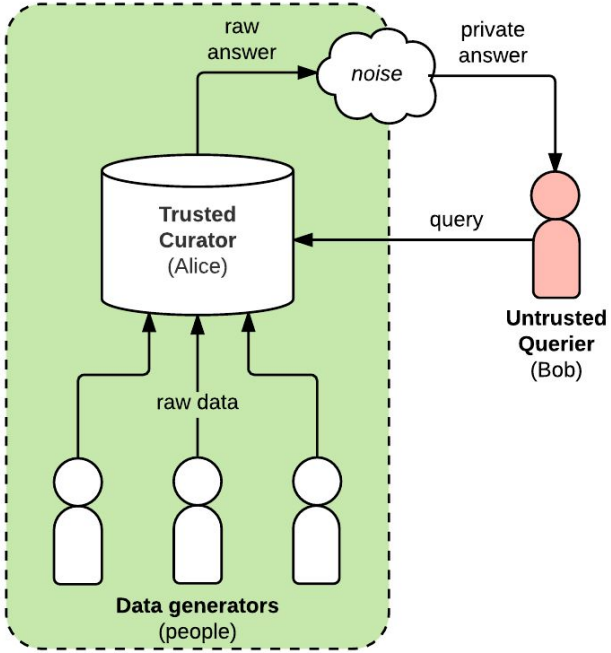
— Cynthia Dwork, *The Algorithmic Foundations of Differential Privacy (2014)* [text]

Defining Differential Privacy

- **local vs. central (global)**



Local privacy



Global privacy

Using Differential Privacy
Shaistha Fathima (2021) [Neptune.ai blog]

Defining Differential Privacy

- **centralized differential privacy (CDP)** assumes a trusted data collector who can collect and access the raw user data
  - the randomized algorithm is applied on the collected dataset to produce differentially private output for downstream use

Defining Differential Privacy

- **local differential privacy (LDP)** does *not* assume a trusted data collector (e.g. users do not want service providers to access and collect their raw text messages)
  - instead, a randomized algorithm is applied on each individual data entry to provide **plausible deniability** before sending it to the untrusted data collector
  - e.g. each user privatizes the text messages before uploading them to the service providers

Applying Differential Privacy to NLP

- **add noise** to text by perturbing tokens
  - (Qu et al., 2021)

Table 1: Examples of perturbed sentences with different choices of η. "Orig" denotes the original input sentence. The red color denotes tokens that are modified.

η	Sequence								
Orig	the	emotions	are	raw	and	will	strike	a	nerve
50	##ori	backward	og	wanda	big	disposal	##pose	lou	##bular
75	410	truth	go	mole	gone	will	strike	y	gifford
100	fine	abused	are	primitive	it	will	slaughter	us	nerve
125	measure	emotions	:	shield	and	relation	strike	nearly	nerve
150	the	caleb	are	kill	and	will	strike	circle	nerve
175	the	emotions	are	raw	and	will	strike	a	nerve

Table 2: Examples of perturbed tokens ($\eta = 100$, sampled independently and sorted by empirical frequencies).

	the	emotions	are	raw	and	will	strike	a	nerve
Perturbed tokens	the	emotions	are	raw	and	will	strike	a	nerve
	a	emotion	were	smackdown	or	would	strikes	the	rebels
	its	emotionally	is	matt	but	can	attack	an	reason
	and	hormones	being	##awa	,	may	drop	his	cells
	his	organizations	re	unused	-	better	##gen	its	spirits
	her	emotional	have	division	as	must	aim	her	bothering
	some	moods	of	protection	"	self	stroke	one	communications

Defining Differential Privacy

- **local differential privacy (LDP)**

Both variants employ a randomized mechanism $M : \mathcal{X} \rightarrow \mathcal{Y}$ that takes in a single data instance $x \in \mathcal{X}$ and outputs a randomized output $y \in \mathcal{Y}$.

LDP [13] requires M to satisfy, for any two inputs $x, x' \in \mathcal{X}$,

$$\frac{\Pr[M(x) = y]}{\Pr[M(x') = y]} \leq e^{\varepsilon}, \forall y \in \mathcal{Y}, \quad (1)$$

where $\varepsilon \geq 0$ is a privacy parameter. Intuitively, Eq. 1 suggests the output of $M(x)$ and $M(x')$ have very similar distributions such that an adversary cannot tell whether the input is x or x'.

Defining Differential Privacy

- **local differential privacy (LDP)**
  - a very strong privacy standard!
    ⇒ regardless of how unrelated *x* and *x'* are, LDP requires them to have similar and indistinguishable output distributions
  - **problem:** as a result of this, the output may not preserve enough information from the original information
    ⇒ this may hurt the **utility** of downstream tasks

Defining Differential Privacy

- ***d* χ-privacy** (also called **Metric-LDP**, or **MLDP**) — a relaxation of LDP
  - allows the indistinguishability of the output distributions **to be scaled** by the distance between the respective inputs

    ⇒ ε in LDP becomes $\eta\, d(x, x')$

***d*χ-privacy [7]**, a relaxation of LDP, was introduced to address the problem mentioned above. More formally, $d\chi$-privacy requires, for any two inputs $x, x' \in \mathcal{X}$,

$$\frac{\Pr[M(x) = y]}{\Pr[M(x') = y]} \leq e^{\eta d(x,x')}, \forall y \in \mathcal{Y}, \tag{2}$$

where $d(x, x')$ is a distance/metric function (e.g., Euclidean distance) and $\eta \geq 0$ is the privacy parameter.[1]

Defining Differential Privacy

- ***d χ*-privacy**
  - allows randomized mechanism *M* to produce more similar output for similar *x* and *x'* measured by *d*(*x, x'*) and thus could preserve more information from the input
    ⇒ note that *η* is the **privacy parameter**, where a smaller *η* promotes "more" privacy by producing a smaller multiple of the distance *d*
  - **caution:** need to measure and calibrate the level of privacy protection — now it depends on the choice of the underlying metric *d*
    ⇒ privacy proof is given in Feyisetan et al. (2020)

Motivation

Differential Privacy for NLP

Motivation for Differential Privacy

- to protect the privacy of users (and other data providers)
  - think about the relationship between users and service providers
    ⇒ in industry, this also affects model deployments (on-client vs. server-side)
  - data breaches!

Privacy Is Important!

- General Data Protection Regulation **(GDPR)** for companies doing business in Europe
  - regulates collecting, storing, utilizing, and transferring customer data

- California Consumer Privacy Act **(CCPA)** in the US
  - people have the right to request companies to disclose the data they store and ask for its removal

Examples of Privacy Attacks

- “Ethical Challenges in Data-Driven Dialogue Systems” (Henderson et al., 2018)
  - exploiting information leakage from in-home virtual assistant devices

> . With conversational models susceptible to such vulnerabilities being integrated into in-home devices, the threat of model exploitation can grow. Let us consider the case of accidental information leakage in such a system. For example, the dialogue agent accidentally records a user revealing private information during a conversation (e.g. “Computer, turn on the lights – *answers the phone* – Hi, yes, my password is...”). If this information is then to used train a conversational model, an attacker can attempt to elicit sensitive information by exploiting the encoder-decoder nature of a dialogue system.

Examples of Privacy Attacks

- **Carlini et al. (2020)[1]:** recover texts from a single document of the training data via querying to an LM pretrained from it
  - extracts verbatim texts of training data
- **Pan et al. (2020)[2] and Song and Raghunathan (2020)[3]:** target the text embedding to reveal data from an encoded query to an NLP service
  - extracts data from queries

[1]Carlini et al. (2020) “Extracting training data from large language models.” [paper]
[2]Pan et al. (2020) “Privacy risks of general-purpose language models.” [paper]
[3]Song and Raghunathan (2020) “Information leakage in embedding models.” [paper]

Previous Work

on Privacy-Preserving Learning for NLP

Approaches to Data Privacy

- **federated learning:** train an ML model on data that is stored on different devices or servers, without having to centrally collect the entirety of the data

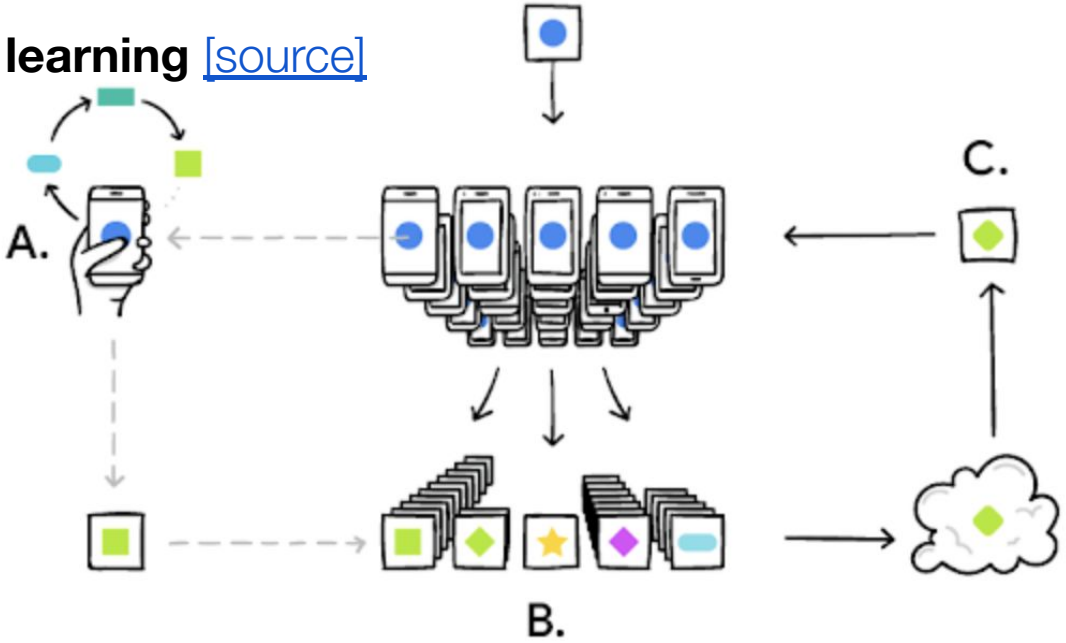
Federated Learning: Collaborative Machine Learning without Centralized Training Data

Thursday, April 6, 2017

Posted by Brendan McMahan and Daniel Ramage, Research Scientists

Approaches to Data Privacy

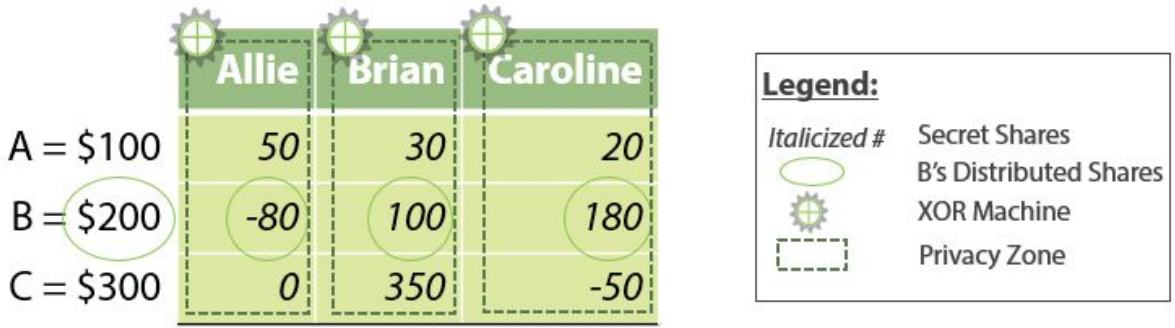
- **federated learning** [source]



Your phone personalizes the model locally, based on your usage (A). Many users' updates are aggregated (B) to form a consensus change (C) to the shared model, after which the procedure is repeated.

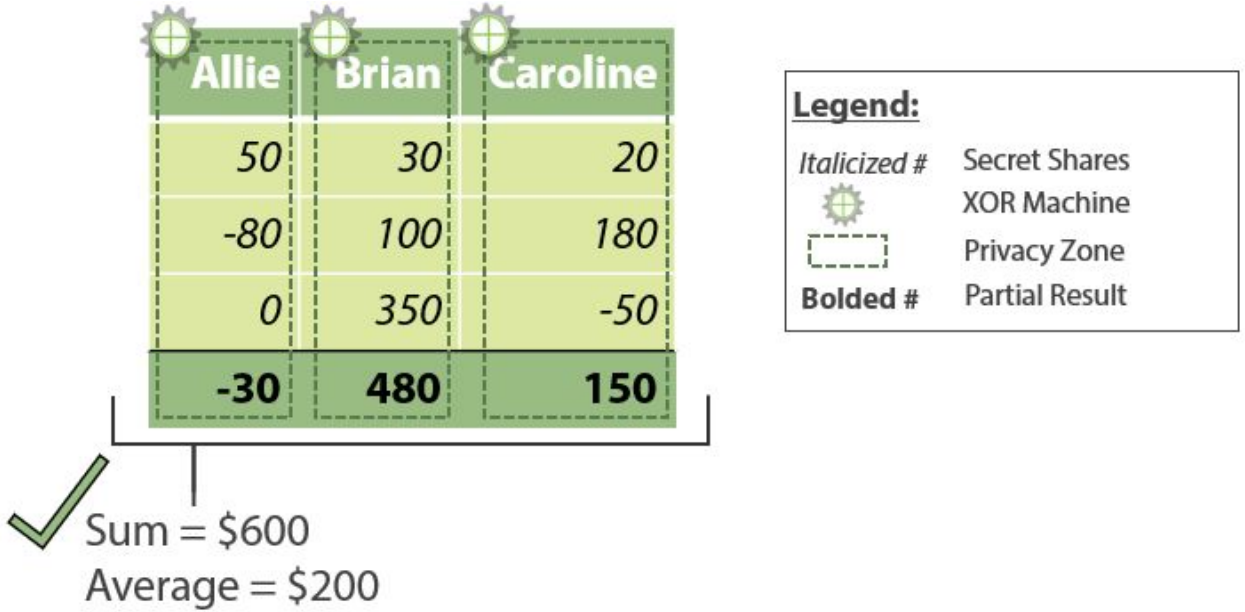
Approaches to Data Privacy

- **homomorphic encryption:** allows you to encrypt your data but still perform computations on it — without decrypting it first
- **secure multi-party computation (SMPC):** distributes a computation across multiple parties, where no individual party can see the other party's data
  - simple example of computing average salary with SMPC [source]



Approaches to Data Privacy

- simple example of computing average salary with SMPC [source]



De-Identification

- **de-identification**
  - Li et al. (2019) [paper]
  - redacts personally identifiable information (PII) in the text
  - Amazon Comprehend [article] is a service that can detect and redact PII

Detecting PII in Amazon Comprehend

When you analyze text using Amazon Comprehend real-time analysis, Amazon Comprehend automatically identifies PII, as summarized in the following table.

PII entity category	PII entity types
Financial	BANK_ACCOUNT_NUMBER BANK_ROUTING CREDIT_DEBIT_NUMBER CREDIT_DEBIT_CVV CREDIT_DEBIT_EXPIRY PIN
Personal	NAME ADDRESS PHONE EMAIL AGE

k-Anonymization

- ***k*-anonymization**
  - Bendersky et al. (2017) [paper] and Li et al. (2019) [paper]
  - retains only words (or n-grams) used by a sufficiently large (i.e. *k*) number of users
- issue: could easily leak other sensitive information

Adversarial Training for Private Representations

- Coavoux et al.'s "Privacy-preserving neural representations of text" (EMNLP 2018) [paper] trains the model's representation in an adversarial setting

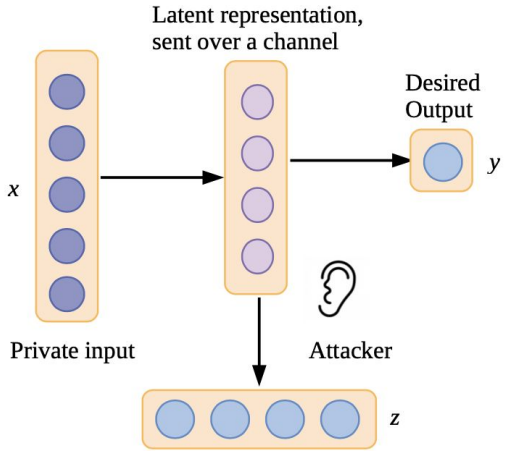


Figure 1: General setting illustration. The main classifier predicts a label y from a text x, the attacker tries to recover some private information $\mathbf{z}$ contained in x from the latent representation used by the main classifier.

Phase 1. Training of the main classifier on (x, y) pairs and evaluation of its accuracy;

Phase 2. Generation of a dataset of pairs $(\mathbf{r}(x), \mathbf{z})$ for the attacker, $\mathbf{r}$ is the representation function of the main classifier ($\mathbf{r}$ is defined in Section 2.1);

Phase 3. Training of the attacker's network and evaluation of its performance for measuring privacy.

Required Paper 1

Natural Language Understanding
with Privacy-Preserving BERT (Qu et al., 2021)

Privacy-Preserving Learning with BERT

- **Qu et al. (2021)'s objectives**
  - guarantee tunable privacy protection at both training and inference time by focusing on token-level privatization
  - implement privacy-adaptive LM pretraining to improve the effectiveness of of pretrained LMs on privatized text

Natural Language Understanding with Privacy-Preserving BERT

Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, Marc Najork
Google
Mountain View, CA, United States
{cqu,weize,yangliuy,mingyang,bemike,najork}@google.com

Privacy-Preserving Learning with BERT

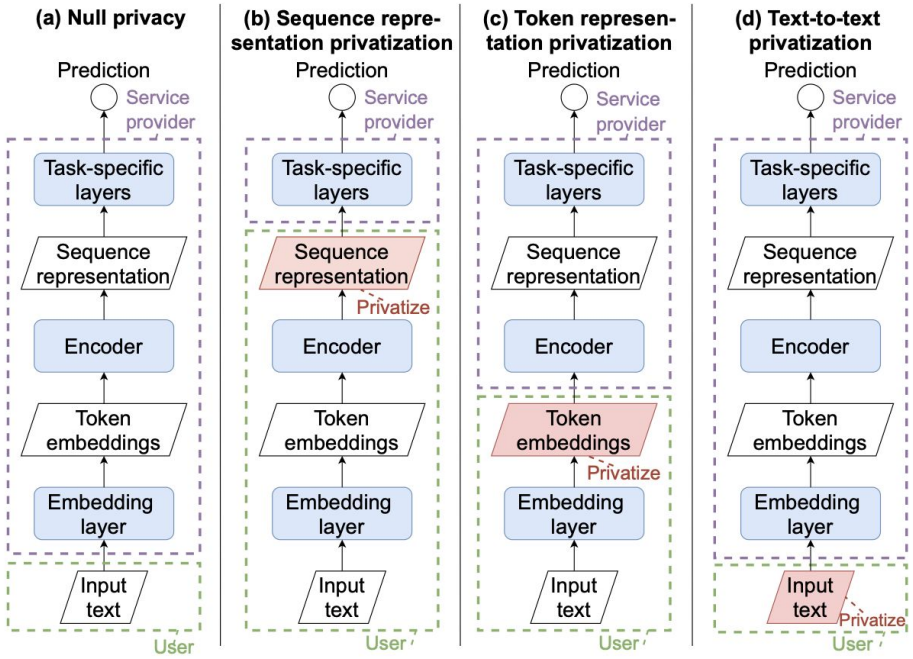


Figure 1: Illustrations of different privacy-constrained methods for training a typical NLU model.

Overview of Approach

- two stages
  - **privacy-preserving mechanism:** each user applies this mechanism to transform raw text to either privatized text or privatized text representations on their own local devices, and then submits the output to the service provider
  - **privacy-adaptive NLU model training:** the service provider can only access the privatized text data for building NLU models

    ⇒ how do we improve NLU performance on privatized text data?

Overview of Approach

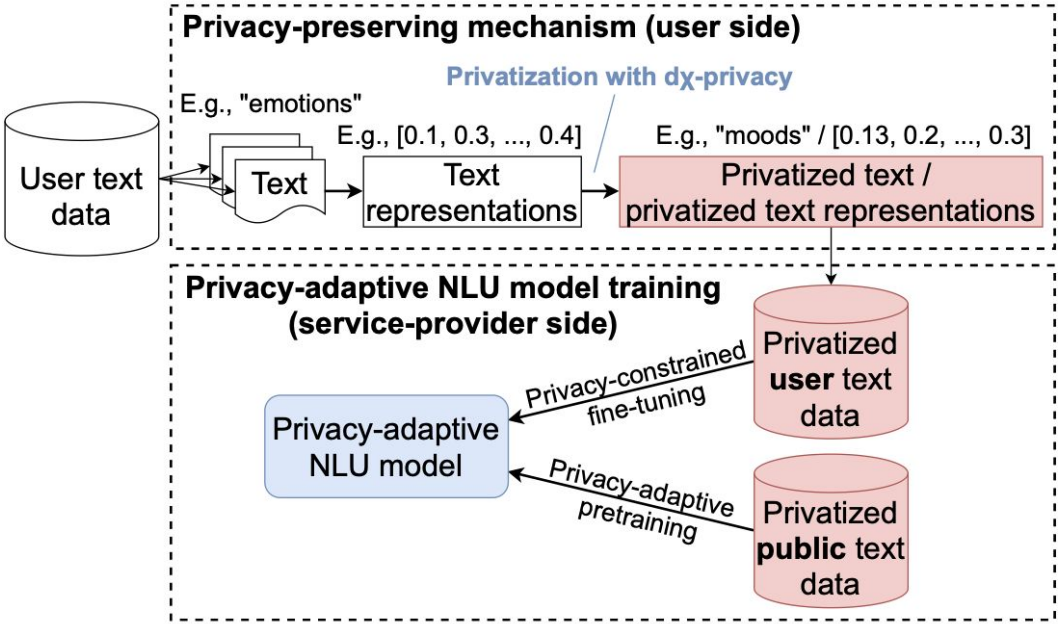


Figure 2: Overview. Users privatize input text locally. Then service providers conduct privacy-adaptive model training.

Overview of Approach

- **improve NLU performance on privatized text data**
  - ⇒ privacy-constrained fine-tuning methods
  - ⇒ privacy-adaptive pretraining methods leveraging large-scale public text corpora

Privacy-Preserving Mechanism

- **token representation privatization** with perturbations

Then $d\chi$-privacy can be achieved for the choice of Euclidean distance $d(x, x') = ||x - x'||$ by adding random noise N drawn from an n-dimensional distribution with density $p(N) \propto \exp(-\eta||N||)$ [10],

$$M_{\text{rep}}(x) = x + N, \tag{3}$$

where we use M_{rep} to denote the privacy mechanism for token representation privatization. This process is referred to as *perturbation* or *noise injection*. To sample N from the noise distribution, consider $N \in \mathbb{R}^n$ as a pair (r, p), where r is the distance from the origin and p is a point in $\mathbb{B}^n$ (the unit hypersphere in $\mathbb{R}^n$). Then we sample $N \in \mathbb{R}^n$ by computing $N = rp$, where r is sampled from Gamma distribution $\Gamma(n, \frac{1}{\eta})$ and p is sampled uniformly over $\mathbb{B}^n$ [11, 33].

Privacy-Preserving Mechanism

- **text-to-text privatization** with nearest neighbor search on perturbations

More specifically, we first embed the input token x using an embedding model $\phi : \mathcal{V} \rightarrow \mathbb{R}^n$. We then pass $\phi(x)$ to M_{rep} to obtain the privatized token representation $M_{\text{rep}}(\phi(x))$. Lastly, we return the token that is closest to $M_{\text{rep}}(\phi(x))$ in the embedding space as the output,

$$M_{\text{txt}}(x) = \arg\min_{w \in \mathcal{V}} ||M_{\text{rep}}(\phi(x)) - \phi(w)||, \quad (4)$$

where M_{txt} denotes the privacy-preserving mechanism for text-to-text privatization.

Privacy-Preserving Mechanism

- **sequence input processing**
  - apply M_{rep} and M_{txt} on each token or token representation to privatize the sequence

Privacy-Constrained Fine-Tuning

- **null privacy:** no privacy constraints → upper bound for model utility
- **sequence representation privatization**
  - embedding layer + encoder are deployed user-side
  - user perturbs sequence representation locally
- **token representation privatization**
  - only embedding layer is deployed user-side
  - user privatizes token embeddings locally
- **text-to-text privatization**
  - users send fully privatized text to the service provider
  - service providers have complete NLU model stack to process it

Privacy-Adaptive BERT Pretraining

- **idea:** we can improve BERT's performance on privatized data by running pretraining on the *privatized versions* of massive amounts of publicly-available unstructured texts
  - doing this makes BERT more robust in handling privatized text or privatized text representations
- **process**
  - initialize model with original BERT checkpoint
  - conduct further pretraining with Next Sentence Prediction (NSP) and **three variants** of Masked LM

Privacy-Adaptive BERT Pretraining

- take advantage of the MLM objective to train the BERT encoder to adapt to the perturbation process more effectively
  - **Vanilla MLM:** predicting the perturbed masked tokens
  - **Probability MLM:** predicting a *set* of perturbed tokens for each masked position
  - **Denoising MLM:** predicting the *original* tokens
    - ⇒ the LM learns to recover the original semantics of the masked token given privatized context
    - ⇒ this, in turn, has some privacy concerns (which we'll look at)

Evaluation

- use two datasets from the GLUE benchmark for both **privacy experiments** and **utility experiments**
  - **Stanford Sentiment Treebank (SST):** sentence classification (sentiment)
  - **Quora Question Pairs (QQP):** sentence-pair classification (paraphrases)

Privacy Experiments

- two experiments are used to demonstrate the level of privacy protection provided by the privacy mechanism on BERT
  - **plausible deniability:** analyze plausible deniability statistics on the vocabulary level
  - **token embedding inversion:** measures the amount of tokens that are modified by perturbations in actual datasets

Privacy Experiments

- **plausible deniability**
  - two statistics are used to characterize the ability of an adversary to recover the original input text when observing the perturbed text or text representations
    - N_w: the probability of an input token w not modified by $M_{\text{txt}}(w)$.
    - S_w: the effective support of the output distribution of perturbation $M_{\text{txt}}(w)$ on the entire vocabulary for an input token w.
  - good privacy guarantees should be characterized by relatively small N_w and relatively large S_w (i.e. large number of unique output tokens)

Privacy Experiments

- **plausible deniability**
  - as η increases (i.e. as noise becomes smaller)
    - ⇒ N_w increases (i.e. tokens have greater probability of being unmodified)
    - ⇒ S_w decreases (i.e. limited support on the vocabulary)

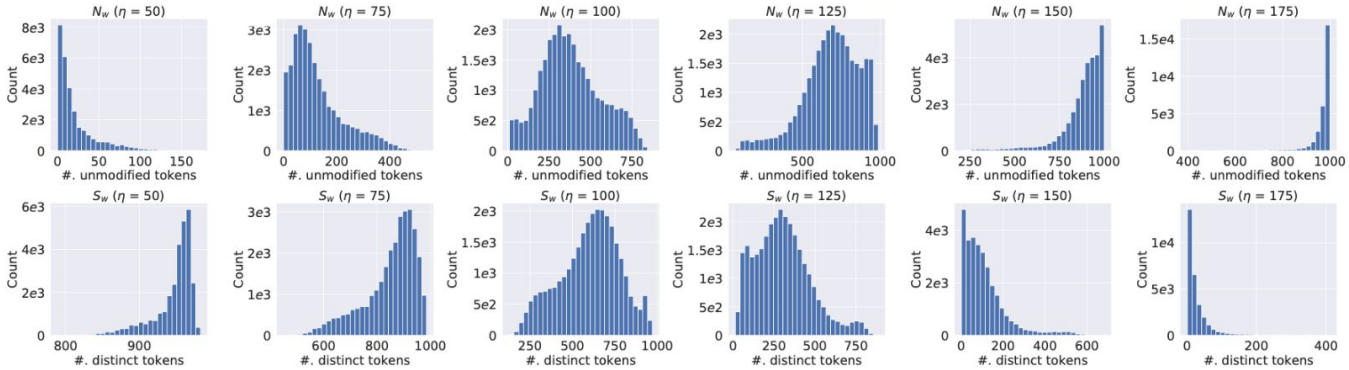


Figure 4: Plausible deniability statistics. Each token in the vocabulary is perturbed 1,000 times. N_w refers to the number of output tokens that are identical to the input token, and S_w refers to the number of unique output tokens.

Privacy Experiments

- **token embedding inversion**
  - measuring the amount of tokens that are modified by perturbation in actual datasets
  - this is an **adversarial task** in which we recover the original tokens based on perturbed token embeddings
    - ⇒ given a perturbed token embedding, find the nearest neighbor of this embedding in the embedding space as the predicted original token
    - ⇒ performance on task is measured by accuracy

Privacy Experiments

- **token embedding inversion**
  - this attack is not comprehensive, but can reveal the privacy–utility tradeoff (and thus help select the privacy parameter η)
  - the results below demonstrate that privacy protection levels are consistent across datasets (and recall that larger η means smaller noise) → the privacy protection mechanism is task- and dataset-agnostic

Table 3: Accuracy of token embedding inversion.

η	50	75	100	125	150	175
SST	0.0154	0.1084	0.3402	0.6354	0.8500	0.9511
QQP	0.0165	0.1066	0.3420	0.6462	0.8620	0.9570

Recall: Privacy-Constrained Methods

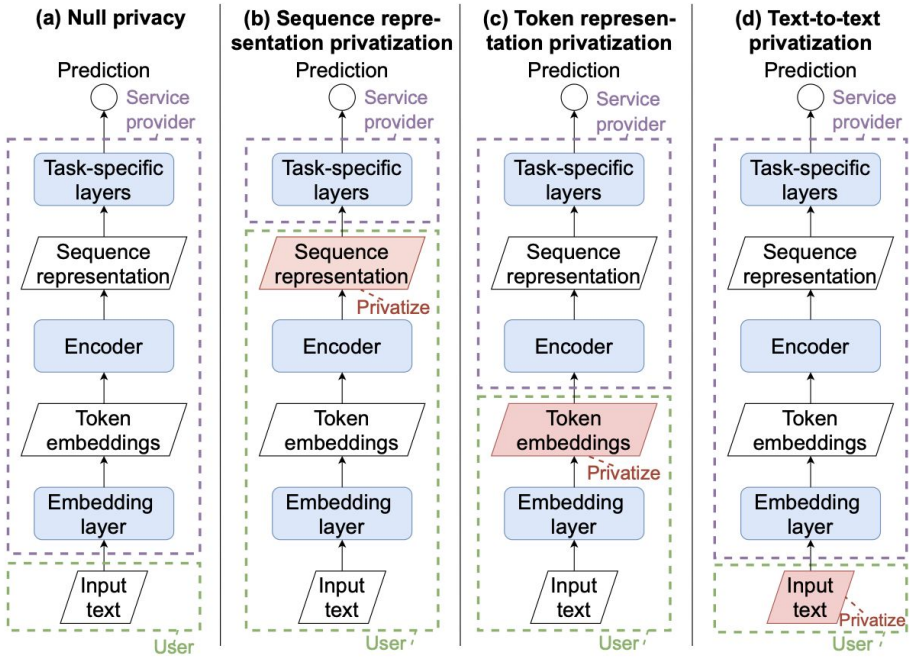


Figure 1: Illustrations of different privacy-constrained methods for training a typical NLU model.

Utility Experiments

- **experiments on sequence representation privatization:** even the smallest perturbations caused a 30% absolute performance decrease compared to null privacy → focus on **token-level privatization** instead (i.e. token representation or text-to-text)
- first, we'll look at the results without any privacy-adaptive pretraining

Utility Experiments

- **SST**

Table 4: Accuracy of privacy-constrained fine-tuning on SST. Scores in the header are obtained with Null Privacy. Boldface denotes better results with the same encoder. Underscores denote better results between different encoders with the same privacy constraint. $\blacktriangle i$ denotes the improvement with respect to column i has statistically significance with $p < 0.05$ tested by the Student's paired t-test.

η	BERT (0.9289)		BiLSTM (0.8406)	
	1. Token Rep	2. Text-to-Text	3. Token Rep	4. Text-to-Text
50	**0.5126**	0.4920	**0.5092**	**0.5092**
75	0.5310	**0.5906**$^{\blacktriangle 1,4}$	**0.5447**	0.5356
100	0.5298	**0.7030**$^{\blacktriangle 1,4}$	0.5608	**0.6697**$^{\blacktriangle 3}$
125	0.5608	**0.8360**$^{\blacktriangle 1,4}$	0.5608	**0.7420**$^{\blacktriangle 3}$
150	0.5665	**0.8968**$^{\blacktriangle 1,4}$	0.5917	**0.8119**$^{\blacktriangle 3}$
175	0.5975	**0.9151**$^{\blacktriangle 1,4}$	0.6216	**0.8211**$^{\blacktriangle 3}$

Utility Experiments

- **QQP**

Table 5: Accuracy of privacy-constrained fine-tuning on QQP. Refer to Tab. 4 to interpret the notations.

η	BERT (0.9106)		BiLSTM (0.8261)	
	1. Token Rep	2. Text-to-Text	3. Token Rep	4. Text-to-Text
50	**0.6370**▲^{2}	0.6318	0.6409▲^{1}	**0.6423**▲^{2}
75	0.6318	**0.6485**▲^{1}	0.6318	**0.6631**▲^{2,3}
100	0.6318	**0.7238**▲^{1,4}	0.6318	**0.7108**▲^{3}
125	0.6318	**0.8274**▲^{1,4}	0.6318	**0.7534**▲^{3}
150	0.6447	**0.8759**▲^{1,4}	0.6811▲^{1}	**0.7854**▲^{3}
175	0.6354	**0.8976**▲^{1,4}	0.6987▲^{1}	**0.8066**▲^{3}

Utility Experiments

- results from experiments without pretraining
  - text-to-text privatization has better performance than token representation privatization
    ⇒ the performance gain is larger when the noise is smaller (i.e. when η is larger)
  - BERT suffers from a larger performance degradation than BiLSTM with token representation privatization

Utility Experiments

- with privacy-adaptive pretraining
  - token representation privatization → Denoising MLM leads

η	SST			
	1. Orig BERT	2. Vanilla MLM	3. Prob MLM	4. Denoising MLM
50	0.5126	*0.5390*$^{\blacktriangle 1}$	***0.5424***$^{\blacktriangle 1}$	*0.5356*
75	0.5310	*0.5791*$^{\blacktriangle 1}$	*0.5757*$^{\blacktriangle 1}$	***0.6089***$^{\blacktriangle 1}$
100	0.5298	*0.6709*$^{\blacktriangle 1}$	*0.6766*$^{\blacktriangle 1}$	***0.7041***$^{\blacktriangle 1,2}$
125	0.5608	*0.7706*$^{\blacktriangle 1}$	*0.7718*$^{\blacktriangle 1}$	***0.7810***$^{\blacktriangle 1}$
150	0.5665	*0.8188*$^{\blacktriangle 1}$	*0.8188*$^{\blacktriangle 1}$	***0.8395***$^{\blacktriangle 1,2,3}$
175	0.5975	*0.8658*$^{\blacktriangle 1}$	***0.8693***$^{\blacktriangle 1}$	***0.8693***$^{\blacktriangle 1}$

Utility Experiments

- with privacy-adaptive pretraining
  - text-to-text privatization → Prob MLM leads (“marginal improvement” over Denoising MLM)

η	SST			
	1. Orig BERT	2. Vanilla MLM	3. Prob MLM	4. Denoising MLM
50	0.4920	*0.5218*	***0.5310***	*0.5092*
75	0.5906	***0.5963***	0.5734	*0.5906*
100	0.7030	*0.7190*	***0.7259***	*0.7225*
125	0.8360	***0.8429***	***0.8429***	*0.8406*
150	0.8968	*0.9048*	***0.9071***	*0.9025*
175	0.9151	*0.9209*	*0.9209*	***0.9220***

Qu et al.’s Conclusions

- BERT is less robust than BiLSTM in handling privatized **token** representations
  - however, BERT pretrained with the Denoising MLM objective is more robust in handling privatized content compared with the original BERT and BiLSTM
- BERT performs better than BiLSTM with **text-to-text** privatization

> To sum up, we recommend adopting Denoising MLM as the primary method for privacy-adaptive pretraining for both token-level privatization approaches. When a relatively strong level of privacy protection (e.g., $\eta < 100$) is required, token representation privatization should be adopted to preserve more utility. Otherwise, text-to-text privatization is preferred. In both cases, privacy-adaptive pretraining is essential to improve model performance.

Required Paper 2

Differential Privacy for Text Analytics
via Natural Text Sanitization (Yue et al., 2021)

Additional Variants of Local Differential Privacy

- recall the definition of local differential privacy **(ε-LDP)**

Definition 1 (ϵ-LDP (Duchi et al., 2013)). *Given a privacy parameter $\epsilon \geq 0$, $\mathcal{M}$ satisfies ϵ-local differential privacy (ϵ-LDP) if, for any $x, x', y \in \mathcal{V}$,*

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(x') = y].$$

Given an observed output y, from the attacker's view, the likelihoods y is derived from x and x' are similar. A smaller ϵ means better privacy due to a higher indistinguishability level of output distributions, yet the outputs retain less utility.

Additional Variants of Local Differential Privacy

- recall the definition of metric LDP **(MLDP)** i.e. *d χ*-privacy

Definition 2 (MLDP). *Given* $\epsilon \geq 0$ *and a distance metric* $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$ *over* $\mathcal{V}$, $\mathcal{M}$ *satisfies MLDP or* $\epsilon \cdot d(x, x')$*-LDP if, for any* $x, x', y \in \mathcal{V}$,

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon \cdot d(x,x')} \cdot \Pr[\mathcal{M}(x') = y].$$

When $d(x, x') = 1 \; \forall x \neq x'$, MLDP becomes LDP. For MLDP, the indistinguishability of output distributions is further scaled by the distance between the respective inputs. Roughly, the effect of ϵ becomes "adaptive."

Additional Variants of Local Differential Privacy

- **utility-optimized LDP (ULDP):** also relaxes LDP, but does so by exploiting the fact that different inputs have different sensitivity levels to achieve higher utility
  - splits the **input** space into **sensitive** and **non-sensitive** sets
  - splits the **output** space into **protected** and **unprotected** sets
    - ⇒ a sensitive input can only be mapped to a protected output (to ensure privacy)
    - ⇒ every output in the unprotected set must be mapped from the nonsensitive inputs (to improve utility)
  - ULDP achieves a guarantee equivalent to LDP for *sensitive* inputs

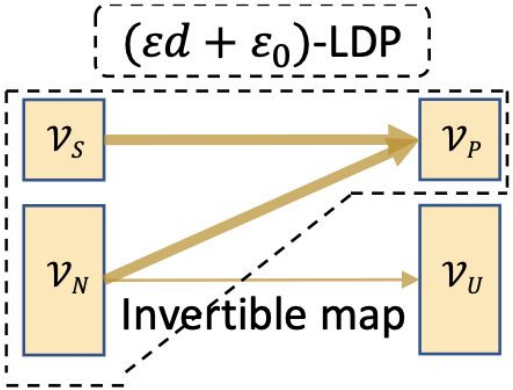
Additional Variants of Local Differential Privacy

- **utility-optimized LDP (ULDP)**

In our context, more formally speaking, let $\mathcal{V}_S \subseteq \mathcal{V}$ be the set of sensitive tokens common to all users, and $\mathcal{V}_N = \mathcal{V} \setminus \mathcal{V}_S$ be the set of remaining tokens. The output space $\mathcal{V}$ is split into the *protected* part $\mathcal{V}_P \subseteq \mathcal{V}$ and the *unprotected* part $\mathcal{V}_U = \mathcal{V} \setminus \mathcal{V}_P$.

Yue et al.'s Differential Privacy Variant

- **utility-optimized metric LDP (UMLDP):** apply metric LDP to only cases where the output is in the protected set



Definition 3 (UMLDP). *Given $\mathcal{V}_S \cup \mathcal{V}_N = \mathcal{V}$, two privacy parameters $\epsilon, \epsilon_0 \geq 0$, and a distance metric $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$, $\mathcal{M}$ satisfies $(\mathcal{V}_S, \mathcal{V}_P, \epsilon, \epsilon_0)$-UMLDP, if*

i) for any $x, x' \in \mathcal{V}$ and any $y \in \mathcal{V}_P$, we have

$$\Pr[\mathcal{M}(x) = y] \leq e^{\epsilon d(x,x') + \epsilon_0} \Pr[\mathcal{M}(x') = y];$$

ii) for any $y \in \mathcal{V}_U$, i.e., from an unprotected set $\mathcal{V}_U$ where $\mathcal{V}_U \cap \mathcal{V}_P = \emptyset$, there is an $x \in \mathcal{V}_N$ such that

$$\Pr[\mathcal{M}(x) = y] > 0,$$
$$\Pr[\mathcal{M}(x') = y] = 0 \; \forall x' \in \mathcal{V} \setminus \{x\}.$$

Yue et al.’s Privacy-Preserving NLP Pipeline

- two **token-wise sanitization methods** with (U)MLDP
  - **SanText:** all tokens are treated as sensitive, which leads to “excessive protection” and utility loss → satisfies MLDP
  - **(Enhanced) SanText+:** applies the UMLDP principle of only protecting the sensitive set of inputs
- injected noise deteriorates performance → two approaches to mitigate this
  - pretrain the encoder
  - fine-tune the full pipeline

SanText: Base Sanitization Mechanism

Algorithm 1: Base Mechanism SANTEXT

Input: A private document $D = \langle x_i \rangle_{i=1}^{L}$, and a privacy parameter $\epsilon \geq 0$

Output: Sanitized document $\hat{D}$

Derive token vectors $\phi(x_i)$ for $i \in [1, L]$;

for $i = 1, \ldots, L$ **do**

  Run $\mathcal{M}(x_i)$ to sample a sanitized token y_i with probability defined in Eq. (1);

end

Output sanitized $\hat{D}$ as $\langle y_i \rangle_{i=1}^{L}$;

SanText+: Enhanced Sanitization

Algorithm 2: Enhanced SANTEXT$^+$

Input: A private document $D = \langle x_i \rangle_{i=1}^{L}$, a privacy parameter $\epsilon \geq 0$, probability p for a biased coin, and sensitive $\mathcal{V}_S$

Output: Sanitized document $\hat{D}$

Derive token vectors $\phi(x_i)$ for $i \in [1, L]$;
for $i = 1, \ldots, L$ **do**
    if $x_i \in \mathcal{V}_S$ **then**
        Sample a substitution $y_i \in \mathcal{V}_P = \mathcal{V}_S$ with probability given in Eq. (1) ▷ Run SANTEXT over $\mathcal{V}_S$ and $\mathcal{V}_P$;
    else
        Output $y_i = x_i$ with prob. $(1 - p)$; or $y_i \in \mathcal{V}_P$ with prob. in Eq. (2);
    end
end
Output sanitized $\hat{D}$ as $\langle y_i \rangle_{i=1}^{L}$;

Improving Downstream Performance

- **pretraining BERT over sanitized public corpora**
  - initialize the encoder with the original BERT checkpoint
  - conduct further pretraining with Denoising MLM objective (i.e. predict the original masked token)

> We note that this is beneficial for improving the task utility, yet may breach the user privacy as the objective learns to "recover" the original tokens or semantics. In Section 5.4, our results will show that such pretrained BERT indeed improves accuracy, with comparable privacy as in original BERT.

Improving Downstream Performance

- **fine-tuning the full NLP pipeline**
  - assume the ground-truth labels are available to the service provider
  - the sanitized text–label pairs are used for training and fine-tuning downstream task models
  - gradients are backpropagated to update the parameters of both the encoder and task layer
  - not fully explored in this paper

Experiments on Downstream Tasks

- three representative downstream NLP tasks (i.e. datasets) with privacy implications
  - **SST-2** (Stanford Sentiment Treeback) for sentiment classification evaluated on accuracy
  - **Med-STS** for **medical semantic textual similarity** (assigns a numerical score to each pair of sentences, indicating the degree of similarity) evaluated on Pearson correlation coefficient
  - **QNLI** for **question natural language inference**

Experiments on Downstream Tasks

- **implementation and results**
  - use sanitized data to train and test prediction models for all three tasks
  - use either GloVe embeddings (to have fair comparison with Feyisetan et al., 2020) or BERT (for the remaining experiments since they have better performance)
  - Table 1 compares results with Feyisetan

Experiments on Downstream Tasks

Mechanisms	SST-2			MedSTS			QNLI		
	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 3$
Random	0.4986	0.4986	0.4986	0.0196	0.0196	0.0196	0.5152	0.5152	0.5152
Feyisetan et al. (2020)	0.5099	0.5143	0.5345	0.0201	0.0361	0.0452	0.5162	0.5256	0.5333
SANTEXT	0.5101	0.5838	0.8374	0.0351	0.5392	0.8159	0.5372	0.5598	0.8116
SANTEXT$^+$	**0.7796**	**0.7943**	**0.8516**	**0.4965**	**0.7082**	**0.8162**	**0.7699**	**0.7760**	**0.8131**
Unsanitized	0.9251			0.8527			0.9090		

Table 1: Utilities comparison of sanitization mechanisms under similar privacy levels using the GloVe embedding

Yue et al.'s Conclusion

Practically, we consider the whole PPNLP pipeline and build in privacy at the root with our sanitization-aware pretraining and fine-tuning. With our simple and clear definition of sensitivity, our work already achieved promising performance. Future research in sophisticated sensitivity measures will further strengthen our approach.

Surprisingly, our PPNLP solution is discerning like a cryptographic solution: it is kind (maintains high utility) to the good but not as helpful to the bad (not boosting up inference attacks). We hope our results with different metrics for quantifying privacy can provide more insights in privacy-preserving NLP and make it accessible to a broad audience.

In Practice

Differential Privacy Tools

Opacus: PyTorch Models with DP

[site] [GitHub]

Opacus

FAILED

Opacus is a library that enables training PyTorch models with differential privacy. It supports training with minimal code changes required on the client, has little impact on training performance and allows the client to online track the privacy budget expended at any given moment.

Opacus: PyTorch Models with DP

[site] [GitHub]

Getting started

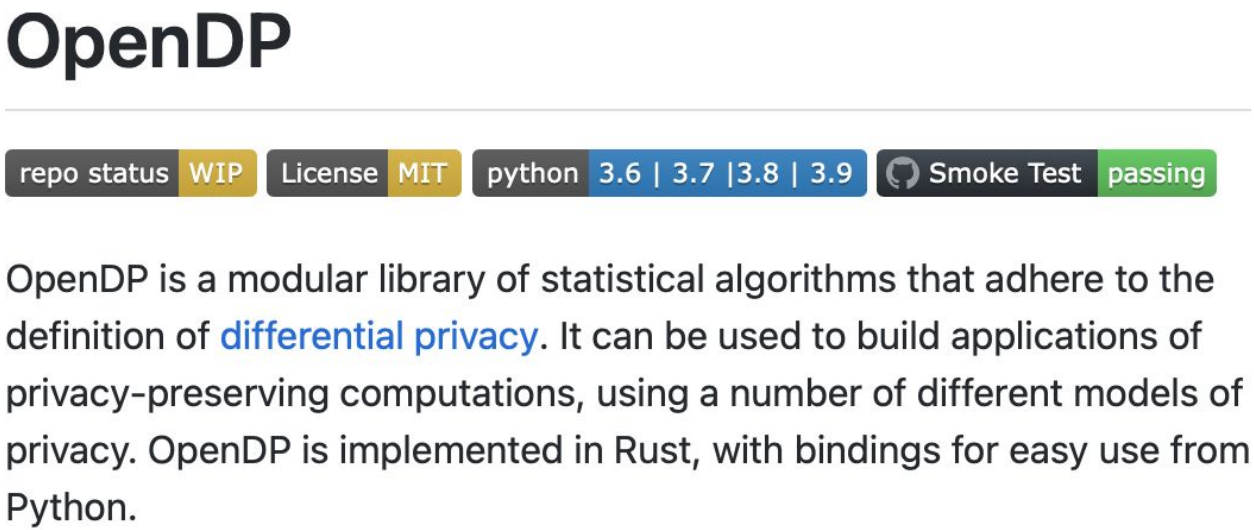
To train your model with differential privacy, all you need to do is to declare a `PrivacyEngine` and attach it to your optimizer before running, eg:

```
model = Net()
optimizer = SGD(model.parameters(), lr=0.05)
privacy_engine = PrivacyEngine(
    model,
    sample_rate=0.01,
    alphas=[10, 100],
    noise_multiplier=1.3,
    max_grad_norm=1.0,
)
privacy_engine.attach(optimizer)
# Now it's business as usual
```

OpenDP

[site] [GitHub]

- Suite of open-source differential privacy tools originally developed by Microsoft and Harvard



Google Differential Privacy

[GitHub]

Differential Privacy

This repository contains libraries to generate ε- and (ε, δ)-differentially private statistics over datasets. It contains the following tools.

- Privacy on Beam is an end-to-end differential privacy framework built on top of Apache Beam. It is intended to be easy to use, even by non-experts.
- Three "DP building block" libraries, in C++, Go, and Java. These libraries implement basic noise addition primitives and differentially private aggregations. Privacy on Beam is implemented using these libraries.
- A stochastic tester, used to help catch regressions that could make the differential privacy property no longer hold.
- A differential privacy accounting library, used for tracking privacy budget.
- A command line interface for running differentially private SQL queries with ZetaSQL.

Conclusion

Conclusion

- Differential privacy is a way to define privacy based on the output distribution of a randomized algorithm (intuitively, a “query”) on two datasets that differ in at most one data entry
- Noise is used to introduce differential privacy
- There is a tradeoff between privacy and utility
- The privatization step can occur at various levels of the NLP pipeline
- Pretraining on privatized corpora can improve language model performance (e.g. with BERT)
- Differential privacy has important implications, particularly in industry, and is an area of active research

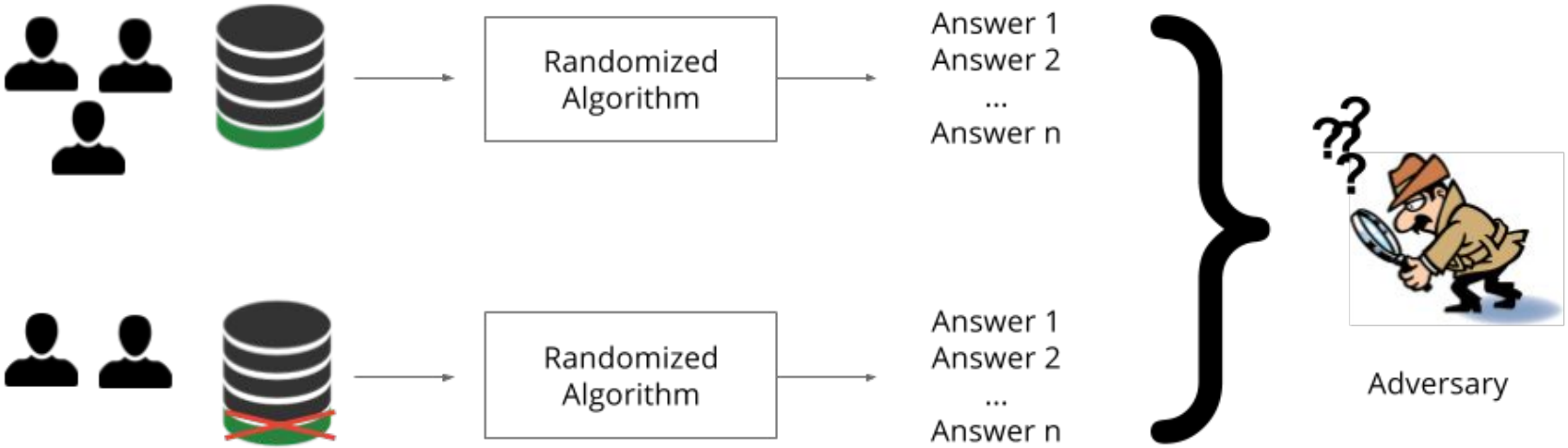
Discussion Questions

Discussion Questions

- How should we incorporate the privacy–utility tradeoff into our projects, both in industry and in academia?
- Will the differential privacy approach of adding noise at various levels of the NLP stack be scalable to all NLP tasks? Are there tasks which cannot incorporate differential privacy?
- We saw both Qu et al. and Yue et al.'s models had performance gains after pretraining on privatized data. Is privacy-adaptive pretraining a feasible solution for huge models like GPT-3? If not, how can we compensate for utility?
- Could differential privacy be used in conjunction with other privacy approaches, like federated learning? What additional considerations would come into play?

Thank you!

Intuition of Differential Privacy



We'll walk through an example with finding the age of a survey participant

Privacy and machine learning: two unexpected allies?
Nicolas Papernot and Ian Goodfellow (2018) [article]