New Threats to Privacy-preserving Text Representations

Huixin Zhan[1], Kun Zhang[2], Chenyi Hu[3] and Victor S. Sheng[1,*]
[1]Department of Computer Science, Texas Tech University
[2]Department of Computer Science, Xavier University of Louisiana
[3]Department of Computer Science, University of Central Arkansas
Email: [1]{huixin.zhan,victor.sheng}@ttu.edu, [2]kzhang@xula.edu, [3]chu@uca.edu
[*]Corresponding author: Victor S. Sheng, victor.sheng@ttu.edu

Abstract

The users' privacy concerns mandate data publishers to protect privacy by anonymizing the data before sharing it with data consumers. Thus, the ultimate goal of privacy-preserving representation learning is to protect user privacy while ensuring the utility, e.g., the accuracy of the published data, for future tasks and usages. Privacy-preserving embeddings are usually functions that are encoded to low-dimensional vectors to protect privacy while preserving important semantic information about an input text. We demonstrate that these embeddings still leak private information, even though the low dimensional embeddings encode generic semantics. We develop two classes of attacks, i.e., adversarial classification attack and adversarial generation attack, to study the threats for these embeddings. In particular, the threats are (1) these embeddings may reveal sensitive attributes letting alone if they explicitly exist in the input text, and (2) the embedding vectors can be partially recovered via generation models. Besides, our experimental results show that our approach can produce higher-performing adversary models than other adversary baselines.

1. Introduction

Textual information is one of the most significant data. Online users generate a great amount of textual information by participating in different online activities such as online reviews and posting tweets. On one hand, textual data consists of abundant information about users' behavior preferences for data consumers to study. On the other hand, publishing complete and intact users' textual data risks exposing privacy information to adversaries. This scenario usually arises when the computation of a neural network is shared across multiple devices, e.g., some hidden representations are computed locally and send to a cloud-based model. In this case, the hidden representations are easy to be obtained by the adversary during uploading the data [Coavoux et al., 2018].

Private information can take the form of key phrases explicitly contained in the text. However, it can also be implicit. For example, demographic information about the author of a text can be predicted with above-chance accuracy from linguistic cues in the text itself [Rosenthal and McKeown, 2011, Preoţiuc-Pietro et al., 2015]. However, letting alone its explicitness, some of the private information correlates with the output labels and therefore can be learned by a neural network as the saying "*you are what you write*" goes.

Recently, a line of privacy-preserving text representation works encode the text inputs by $\Phi(\mathbf{x})$ in order to preserve the rich semantic meaning while not compromising too much utility for various downstream tasks. However, this setting is not applicable for the cloud servers in real-world because (1) it's not possible to ongoingly share the model with the users' devices, (2) it can be computationally intensive for computing all the embedding via $\Phi(\mathbf{x})$ for the local users' device, and (3) manipulating the full dataset increases the chance for being detected by the adversary. Thus, we investigate the threats for the embedding in a more realistic setting, where the server only manipulate a proportion of the training data $\mathcal{D}$ via $\Phi(\mathbf{x})$ and manipulating this set of data is sufficient for preventing the adversary from privacy recovering, while the data are not manipulated (the true embeddings) remain the typical sentence embeddings.

In this setting, even though these embeddings are in a low dimension, do they, perhaps, inadvertently capture the demographic information about the authors in addition to the meaning of words? Would that have consequences if these embeddings are used in adversarial ways? These questions motivate our study on threats for these embeddings. Song and Raghunathan [2020] introduced embedding inversion attacks and attribute inference attacks. These methods assume the adversary has access to Φ by querying a small set of auxiliary data $\mathcal{D}_a$. However, in most cases,

URI: https://hdl.handle.net/10125/79426
978-0-9981331-5-7

the auxiliary data is not ideally drawn from the same underlying process from the training data. Thus, the performance for learning the model is consequently inferior. In order to address this issue, we instead study the case when the adversary can not access this neural network. In particular, we develop two classes of attacks to study information that might be leaked by embeddings. The first is called **adversarial classification (AC)**, where the embeddings might reveal sensitive attributes letting alone if they explicitly exist in the input text. The second is **adversarial generation (AG)**, where the embedding vectors can be partially recovered from the input data via the generation models. We propose the adversary methods based on a semi-supervised generative adversarial network (SGAN). In our proposed SGAN, the adversarial network distinguishes the true embeddings from the manipulated embeddings, and the adversary is trained by a combinatorial loss that leverages both the supervised auxiliary data and the unsupervised training data. The model is then fine-tuned on a testing data (include both $r(\mathbf{x}; \Phi)$ and $r(\mathbf{x}; \theta_r)$ as well) via transfer learning. Our experimental results show that SGAN can produce higher-performing adversary models when comparing with the baseline: inverting the given embeddings back to the sensitive raw text inputs via querying the model.

Our contributions are summarized as follows:

- We investigate the threats for the embedding under a more realistic setting, where the server only manipulates a proportion of the training data via $\Phi(\mathbf{x})$ and manipulating this set of data is sufficient for perturbing the training of the adversary.
- When Φ is not accessible, we propose an SGAN approach, where the adversarial network distinguishes the true embeddings from the manipulated embeddings, and the adversary is only trained based on the true embeddings. Thus, the training of the adversary is not perturbed by the manipulated privacy-preserving embeddings.
- Experimental results show that SGAN produces higher-performing adversary models than other adversary baselines.

2. Related works

We formulate the related works in two parts, the privacy-preserving representation learning, and study the information leakage for the embeddings under the privacy-preserving representation learning scenario.

2.1. Privacy-preserving representation learning

In a line of privacy-preserving representation learning studies, Li et al. [2018] proposed ADV that based on GAN [Goodfellow et al., 2014] to improve the robustness and privacy of neural representations. This method is applied to part-of-speech tagging and sentiment analysis. Similar to [Ganin et al., 2016], ADV learns a discriminator model jointly with learning the standard supervised model. However, this method cannot eliminate the difference regarding the training performance for some author characteristics that are more sensitive in contributing to the unforeseen impacts in differing the model performance for different user groups. However, our method can eliminate the performance difference while requiring a certain privacy budget. In contrast, Coavoux et al. [2018] propose multi-detasking and adversarial generation methods. These methods train a new classifier from scratch to evaluate privacy once the parameters of our main model are fixed. Recently, Beigi et al. [2019] proposed DPText that learns a differentially private representation by minimizing the chance of the adversary to infer whether target text representation is in the database with the weighted sum objective. However, all these methods focus on encoding all the text inputs by $\Phi(\mathbf{x})$. However, (1) it's not possible to ongoingly share the model with the users' devices, (2) it can be computationally intensive for computing all the embedding via $\Phi(\mathbf{x})$ for the local users' device, and (3) manipulating the full dataset is easy to be detected by the adversary. Thus, we investigate the threats in a more realistic setting and the experimental results perform higher-performing adversary models than the embedding inversion attacks [Song and Raghunathan, 2020].

2.2. Attacking the privacy-preserving representation learning

A line of works [Song and Shmatikov, 2019, Song and Raghunathan, 2020] leverage attribute inference attacks to test if embeddings might unintentionally reveal attributes of their input data that are sensitive. This threat model is particularly important when sensitive attributes orthogonal to downstream tasks are quickly revealed given very little auxiliary data. However, these methods assume the adversary has access to Φ by querying a small set of auxiliary data. However, in most cases, the auxiliary data are not ideally drawn from the same underlying process from the training data, which leads to inferior training performance. In addition, the authors investigate

membership inference attacks as well. However, our paper is not focusing on addressing this problem.

3. Methods

In this section, we introduce the adversarial scenario, the basic defense and attack methods, and the proposed SGAN approach.

3.1. The Adversarial Scenario

First, we propose to frame the training of the main classifier as the cloud server via a two-agent process: (1) the main agent and (2) an adversary. They are exploiting a setting similar to GAN, where the generator learns to reconstruct privacy information from the low-dimensional embeddings, whereas the main agent learns to perform its main task and to make the adversary difficult in recovering the privacy. In the adversarial scenario, each example consists of a triple $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, where $\mathbf{x}$ is a natural language text for each data instance, $\mathbf{y}$ is a single label, e.g. topic or sentiment, and $\mathbf{z}$ is a vector of private information contained in $\mathbf{x}$. In our two agents setting, (i) the adversarial generator learns to predict $\mathbf{z}$ from the latent representation $r(\mathbf{x})$ of $\mathbf{x}$ used by the main classifier, and (ii) the main agent learns to predict $\mathbf{y}$ from $\mathbf{x}$. $r(\mathbf{x})$ is composed of two parts of the training data: the true embeddings $r(\mathbf{x}; \theta_r)$ and the manipulated embeddings $r(\mathbf{x}; \Phi)$.

In order to evaluate the utility, e.g., accuracy and privacy of a specific model, we proceed in four phases as shown in Figure 1:

- Phase 1. Training of the main classifier on $(\mathbf{x}, \mathbf{y}, \mathbf{z})$ in adversarial classification case or $(\mathbf{x}, \mathbf{y})$ pairs only in adversarial generation case and evaluation of its accuracy;
- Phase 2. Generation of a dataset of pairs $(r(\mathbf{x}), \mathbf{z})$ for the adversary, r is the representation function of the main classifier;
- Phase 3. Training of the adversary's network on $(r(\mathbf{x}), \mathbf{z})$ in adversarial classification case or training of the adversary on $r(\mathbf{x})$ in adversarial generation case;
- Phase 4. Evaluation of the adversary's performance for measuring privacy.

3.2. Basic Defense Methods

Here we propose two types of basic defense methods. The first is to defend the adversarial classification (DAC) and the second one is to defend

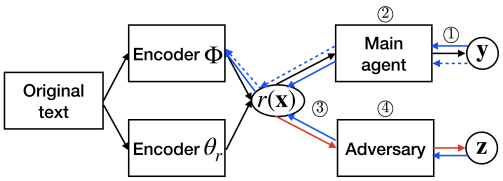


Figure 1: The main agent predicts a label $\mathbf{y}_i$ from a text $\mathbf{x}_i$, the adversary tries to recover some private information $\mathbf{z}_i$ contained in $\mathbf{x}_i$ from the embeddings (in red). All back propagations for adversarial classification are shown in blue. All back propagations for adversarial generation are shown in blue dashed line.

the adversarial generation (DAG). We will then reveal the threats for both DAC and DAG. Note that in both methods, we all first randomly select a small K proportion of manipulated data from the dataset (including the training set, the validation set, and the test set), where $K = \frac{\text{the number of manipulated data}}{\text{the number of all data}}$. All the objectives are only trained on Φ. Other data are not taking part in the training of the parameter Φ, whereas trained by normal sentence embeddings as follows.

Sentence embeddings are functions that map a variable-length sequence of words $\mathbf{x}$ to a fix-sized embedding vector $r(\mathbf{x}, \theta_r)$ through a neural network model θ_r. For a input sequence of n words $\mathbf{x} = [w_1, w_2, ..., w_n]$, θ_r first maps $\mathbf{x}$ into a sequence of word vectors $\mathbf{x} = [v_1, ..., v_n]$ with a word embedding matrix v such as Glove [Pennington et al., 2014]. Then θ_r feeds $\mathbf{x}$ to a recurrent neural networks (rnn) [Mikolov et al., 2010] or Transformer [Jaderberg et al., 2015] and obtain a sequential hidden representation $[h_1, h_2, ..., h_n]$ for each word in $\mathbf{x}$. Finally θ_r outputs the sentence embedding by reducing the sequential hidden representation to a single vector representation.

The manipulated embeddings are trained as follows:

DAC The main classifier optimizes:

$$\mathcal{L}_m(\mathbf{x}, \mathbf{y}, \mathbf{z}; \Phi, \theta_r, \boldsymbol{\theta_p}) = -\alpha \log P(\mathbf{y}|\mathbf{x}; \Phi, \theta_r, \boldsymbol{\theta_p}) - \beta \log P(\neg \mathbf{z}|r(\mathbf{x}); \boldsymbol{\theta_p}), \quad (1)$$

where $\boldsymbol{\theta_p}$ represents the parameters for the main classifier, and Φ represents the parameters for the manipulated data. The first term of this equation is to minimize the negative log-likelihood of the $\mathbf{y}$ labels, and the second term is designed to deceive the adversary. Both $\alpha > 0$ and $\beta > 0$ control the relative importance of both terms.

DAG Similarly, the loss of the main model is as follows:

$$\mathcal{L}_m(\mathbf{x}, \mathbf{y}; \Phi, \theta_r, \boldsymbol{\theta_p}) = -\alpha \log P(\mathbf{y}|\mathbf{x}; \Phi, \theta_r, \boldsymbol{\theta_p}) - \beta(-\mathcal{L}'_a(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta'_a}, \Phi, \theta_r)), \tag{2}$$

where $\boldsymbol{\theta'_a}$ represents the parameters for the adversary in adversarial generation, the first term minimizes the negative likelihood of the $\mathbf{y}$ labels whereas the second term is meant at making the recovering difficult by maximizing the loss of the generator, $\mathcal{L}'_a$ is defined in subsection 3.3.

3.3. Basic Adversarial Attacks

We experiment with two types of adversaries: a classifier that predicts the binary attributes $\mathbf{z}(\mathbf{x})$ used as a proxy for the reconstruction of $\mathbf{x}$ and a character-based language generation model that directly optimizes the likelihood of the training examples. In the adversarial attacks, we generalize a dataset made of pairs $(r(\mathbf{x}), \mathbf{z}(\mathbf{x}))$, where $r(\mathbf{x})$ is the embedding from the main classifier and $\mathbf{z}(\mathbf{x})$ is a vector of private categorical variables. In our experiments, we use the same training examples $\mathbf{x}$ for both the main classifier and the adversary. The two adversaries are as follows:

Adversarial Classification Formally, for a single data point $(\mathbf{x}, \mathbf{y}, \mathbf{z})$, the adversarial classifier optimizes:

$$\mathcal{L}_a(\mathbf{x}, \mathbf{y}, \mathbf{z}; \boldsymbol{\theta_a}, \Phi, \theta_r) = -\log P(\mathbf{z}|r(\mathbf{x}); \boldsymbol{\theta_a}). \tag{3}$$

Once the main model has been trained, the parameters θ_r are fixed. $\boldsymbol{\theta}_a$ represents the parameters for the adversarial classification. As in a GAN, the losses of both classifiers are interdependent, but their parameters are distinct: the adversary can only update $\boldsymbol{\theta_a}$ or $\boldsymbol{\theta'_a}$ and the main classifier can only update θ_r and $\boldsymbol{\theta_p}$. When we train the main model, the weights for the adversary are fixed.

Adversarial Generation The second type of generator is a character-based LSTM language model that is trained to reconstruct full training examples. For a single data point, the hidden state of the LSTM is initialized with $r(\mathbf{x})$, computed by the main model. The generator optimizes:

$$\mathcal{L}'_a(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta'_a}, \Phi, \theta_r) = -\sum_{c=1}^{C} \log P(r(\mathbf{x}_c)|r(\mathbf{x}_{c-1}); \boldsymbol{\theta'_a}), \tag{4}$$

where $\mathbf{x}_i$ is the i-th character in the document, and C is the length of the document in number of characters. The generator has no control over $r(\mathbf{x})$, and optimizes the objective only by updating its own parameters $\boldsymbol{\theta'_a}$.

3.4. The proposed SGAN approach

After we have introduced the basic defense strategy, we will then move to our proposed SGAN approach. For the adversary, our aim is recovering as much the private information as possible given the possibly manipulated input embeddings.

First, we recall that the recovering strategy of the basic adversaries is to recover as much the private information $\mathbf{z}$ as possible, when the possibly manipulated embedding $r(\mathbf{x})$ are given. The loss is to minimize the negative log-likelihood of the private information $\mathbf{z}$ as shown in this formula:

$$\mathcal{L}(\theta) = \sum_{i=1}^{N} -\log P(\mathbf{z}|r(\mathbf{x}); \theta), \tag{5}$$

where $\mathcal{L}$ can be either $\mathcal{L}_a$ or $\mathcal{L}'_a$, there are totally N data instances, θ are the set of all trainable parameters $\{\Phi, \boldsymbol{\theta_p}\}$. The problem for the basic adversary is, when given the possibly manipulated $r(\mathbf{x})$, it is hard for the adversary to recover any relevant private information $\mathbf{z}$.

We follow the principle that if θ is trained on the manipulated data, it's hard for the adversary to recover any useful information, but if we can identify which part of the embeddings are manipulated and which part of the data are not, the training can be only conducted on the true embeddings, and the recovering ability for the adversary will not be a problem. In order to address this, we propose an SGAN approach, where the adversarial network distinguish the true embeddings from the manipulated embeddings. Semi-supervised learning needs a few labeled examples from the auxiliary dataset $\mathcal{D}_a$ and many unlabeled examples. We can get the small set of labeled examples from an external domain. The true and manipulated embeddings for the small labeled set can be obtained by training $\boldsymbol{\theta_s}$ via the positive and the negative loss in (4). In other words, we introduce another neural network $\boldsymbol{\theta_s}$ for distinguishing the true embeddings from the manipulated embeddings, the labels $\mathbf{y_s}$ for $\boldsymbol{\theta_s}$ is binary, i.e., either true ($\mathbf{y_s} = 1$) or false ($\mathbf{y_s} = 0$).

Here we introduce the SGAN approach, because we have both the labeled data and unlabeled data, our loss function for training the GAN then becomes the supervised loss $\mathcal{L}_{su}$ plus the unsupervised loss $\mathcal{L}_{un}$:

$$\mathcal{L}_{\text{sum}}(\boldsymbol{\theta_s}) = \mathcal{L}_{\text{su}} + \mathcal{L}_{\text{un}}, \tag{6}$$

where

$$\mathcal{L}_{\text{su}}(\boldsymbol{\theta_s}) = \\ -\mathbb{E}_{\mathbf{x},\mathbf{y_s}\sim\mathcal{D}_a(\mathbf{x},\mathbf{y_s})} \log P(\mathbf{y}|\mathbf{x},\mathbf{y_s}=1;\boldsymbol{\theta_s}), \quad (7)$$

and

$$\mathcal{L}_{\text{un}}(\boldsymbol{\theta_s}) = -\mathbb{E}_{\mathbf{x}\sim\mathcal{D}} \log[1-P(\mathbf{y_s}=0|\mathbf{x};\boldsymbol{\theta_s})] \\ -\mathbb{E}_{\mathbf{x}\sim\mathcal{D}} \log[P(\mathbf{y_s}=0|\mathbf{x};\boldsymbol{\theta_s})]. \quad (8)$$

For the supervised loss, we have decomposed the total cross-entropy loss into its loss function, which is the negative log probability of the label, given that the true embedding. For the unsupervised loss, we recall that the approach for the standard GAN is shown in (9). We maximize the probability for the true embedding is classified as true, and we minimize the probability for the generated data $\mathbf{g}$ is true, because the generator G is against the discriminator $\boldsymbol{D}$. We also know that $\boldsymbol{D}(\mathbf{x}) = 1 - P(\mathbf{y_s}=0|\mathbf{x})$. Thus, by substituting $\boldsymbol{D}(\mathbf{x})$ with $1-P(\mathbf{y_s}=0|\mathbf{x})$, the unsupervised loss for (8) is in fact the standard GAN game-value as becomes evident when we substitute $\boldsymbol{D}(\mathbf{x}) = 1 - P(\mathbf{y_s}=0|\mathbf{x})$ into the expression:

$$\mathcal{L}_{\text{un}} = -\mathbb{E}_{\mathbf{x}\sim\boldsymbol{D}(\mathbf{x})} \log[\boldsymbol{D}(\mathbf{x})] \\ -\mathbb{E}_{\mathbf{g}\sim\text{noise}} \log(1-\boldsymbol{D}(G(\mathbf{g}))). \quad (9)$$

The only difference is, for the SGAN, we do not introduce a generator G for generating new embeddings, we consider all the generated data are coming from the batch training of $\mathcal{D}$ instead, since it includes the true embeddings $r(\mathbf{x};\theta_r)$ and the manipulated embeddings $r(\mathbf{x};\Phi)$ already. The full scenario is shown in Figure 2.

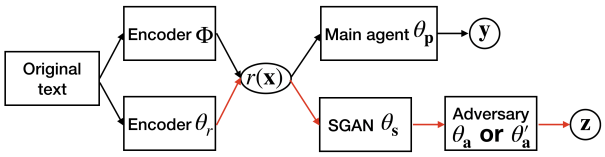


Figure 2: General setting illustration. The red lines show that only the true embeddings $r(\mathbf{x};\theta_r)$ are leveraged to train the adversary. Thus, the parameters for the adversary, i.e., $\boldsymbol{\theta_a}$ or $\boldsymbol{\theta'_a}$ will not be perturbed by the manipulated embeddings.

4. Experimental results

In this section, we discuss the tasks and datasets, settings, and the experimental results on both tasks.

4.1. Tasks and datasets

We conduct our experiment on two tasks. For the first task, the downstream task for the main classifier is sentiment analysis while the adversary conducts adversarial classification. For the second task, the main classifier conducts text classification while the adversary conducts adversarial generation. Next, we introduce these two tasks:

Sentiment Analysis We use the Trustpilot dataset [Hovy et al., 2015] for sentiment analysis. This corpus contains reviews associated with a sentiment score on a scale of five points. We use the five subcorpora corresponding to five areas, i.e., Denmark, France, Germany, United Kingdom, and United States. We extract examples containing both the birth year and gender of the author of the review and use these as the private information. We classify the age of the author into two categories ('under 35' and 'over 45') based on a previous work [Hovy et al., 2015]. Finally, we randomly split each subcorpus into a training set (80%), a validation set (10%) and a test set (10%). We consider two types of auxiliary data $\mathcal{D}_a$: same-domain and cross-domain data. For same-domain data, we use a set of 500 randomly sampled examples from the Trustpilot dataset that is disjoint to the test set. For cross-domain data, we use a set of 300 randomly sampled examples from the blog posts dataset [Schler et al., 2006].

We would also like to evaluate whether the adversary's task is easier when the variables to predict are explicitly in the input, compared to when these information are only potentially and implicitly in the input. As an additional experimental setting, we include both gender and age as input to the main model. We do so by adding two additional tokens at the beginning of the input text, one for each private information. It has also been shown that those variables can be used to improve text classification [Hovy et al., 2015]. In the rest of this section, we use RAW to denote the setting where only the raw text is used as input and $+DEMO$ to denote the setting where the two demographic variables are also used as input.

In this case, the task for the main classifier predicts the sentiment analysis results $\mathbf{y}$ from training examples $\mathbf{x}$. $\mathbf{z}$ is a vector of binary variables, representing, e.g., Age or Gender information about the author. The adversary predicts $\mathbf{z}$.

Text Classification We perform privacy data prediction on blog posts. We used the blog authorship corpus presented by [Schler et al., 2006], a collection of blog posts associated with the Age and Gender of

Table 1: Sizes of both datasets in number of examples

Datasets	Training	Validation	Test
Germany	12596	1574	1574
Denmark	82193	10274	10274
France	9136	1141	1141
UK	48647	6080	6080
USA	22142	2767	2767
Blog posts	5144	642	642

the authors, as provided by the authors themselves. Because the blog posts have no topic annotation, we ran the Latent dirichlet allocation (LDA) algorithm [Blei et al., 2003] on the whole collection (with 10 topics). The LDA outputs a distribution on topics for each blog post. We selected posts with a single dominating topic ($> 80\%$) and discarded the other posts. For the main classifier, we binned Age with two classes, under 20 and over 30. The sizes of both datasets in number of examples are shown in Table 1.

For the adversary, we binned Age with two characters $'U'$ and $'O'$, where $'U'$ represents under 20, and $'O'$ represents over 30. We also binned the Gender with two characters $'F'$ and $'M'$, where $'F'$ represents female and $'M'$ represents male. For the 256 characters in total, we model each of the character as an one-hot class label. For this dataset, we concatenate each post with a $< eod >$ token and two characters of the private information. For example, the following Figure 3 shows an example of the blog posts data that we generated. In the testing phase, the LSTM reads through the whole sequence, and start to predict the first output when reading the $< eod >$ token. Then for each prediction step, the $< eod >$ token is fed as the input and the target character index (treated as a class label) is expected as the output. Finally, we split the corpus into a training set (80%), a validation set and a test set (10% each). For $\mathcal{D}_a$, we follow the aforementioned setting.

[Every day should be a half day., $< eod >$, 'U', 'F']

Figure 3: An example about our generated dataset.

In this case, the task for the main classifier predicts the text classification results $\mathbf{y}$ from training examples $\mathbf{x}$. $\mathbf{y}$ is a vector of binary variables, representing e.g., Age or Gender information about the author. $\mathbf{z}$ is the one hot representation for each two private characters at the end of each example. The adversary predicts $\mathbf{z}$ through adversarial generation.

4.2. Settings

Implementation details We implemented our model using Dynet [Neubig et al., 2017]. The feedforward components (both of the main model and of the adversary) have a single hidden layer of 64 units with a ReLU activation. Each character is modeled as an one-hot class label. The word embeddings have 32 units. We used the Adam optimizer [Kingma and Ba, 2014] with the default learning rate, and 0.2 dropout rate for the LSTM. We used $\alpha = \beta = 1$ for the DAC, based on preliminary experiments. For the DAG method, we used $\alpha = \beta = 1$ as well. For each dataset, and each LSTM state dimension ($\{8, 16, 32, 64, 128\}$), we train the main model for 8 epochs (sentiment analysis) or 16 epochs (topic classification), and select the model with the best accuracy on the development set. Then, the adversary model is trained based on the training data $\mathcal{D}$ for 16 epochs.

Baselines We compare our methods with the following three baselines:

- We utilize **adversarial classification (AC) and adversarial generation (AG) without (w.o.) SGAN** as a straightforward baseline.
- We compare our adversarial generation with the **embedding inversion attacks (EIA)** [Song and Raghunathan, 2020], where the adversary's goal is to invert a target embedding $\Phi(\mathbf{x})$ and recover embeddings in $\mathbf{x}$. The adversary can access Φ.
- We compare our adversarial classification with the **Sensitive attribute inference attacks (SAIA)** [Song and Raghunathan, 2020], where an adversary performing the inference by learning a classifier f on $\mathcal{D}_a$ with the access to Φ.

Evaluation metrics For the main classifier, we report a single accuracy measure. For measuring the privacy of an embedding, we use the following metrics:

- For the sentiment analysis: we use X as the accuracy of the adversary on the prediction of gender and age.
- For the text classification: we use F as the F-1 score of the adversary on the prediction of gender and age.

4.3. Experiments on both tasks

In this subsection, we answer three questions as follows: **1)** Does our method achieve the best adversarial

Table 2: Comparisons between best adversaries (via rnn) for AC w.o. SGAN, SAIA, and AC (when $K = 35\%$). All metrics reported in this table are $X(\%)$.

	Best adversaries for AC w.o. SGAN				Best adversaries for SAIA				Best adversaries for AC			
	RAW		*+DEMO*		*RAW*		*+DEMO*		*RAW*		*+DEMO*	
	Gender	Age	Gender	Age	Gender	Age	Gender	Age	Gender	Age	Gender	Age
Germany	24.8	39.6	24.8	41.4	65.7	69.4	66.7	70.1	70.7	74.8	73.1	78.4
Denmark	31.5	24.7	38.0	36.6	73.7	72.4	76.5	73.4	80.1	78.5	83.5	80.2
France	39.0	42.9	39.0	39.4	77.2	79.9	80.4	83.1	84.9	85.1	87.1	89.0
UK	33.6	36.5	40.1	38.2	74.0	75.4	79.1	80.3	83.1	85.0	86.7	88.4
USA	18.7	25.1	35.3	36.1	53.4	55.1	60.0	61.4	60.9	64.5	66.7	67.1

Table 3: Comparisons between best adversaries (via rnn) for AG w.o. SGAN, EIA, and AG (when $K = 35\%$). All metrics reported in this table are $F(\%)$.

	Best adversaries for AG w.o. SGAN				Best adversaries for EIA				Best adversaries for AG			
	RAW		*+DEMO*		*RAW*		*+DEMO*		*RAW*		*+DEMO*	
	Gender	Age	Gender	Age	Gender	Age	Gender	Age	Gender	Age	Gender	Age
Blog	-	-	33.9	42.0	-	-	71.5	79.4	-	-	76.7	83.2

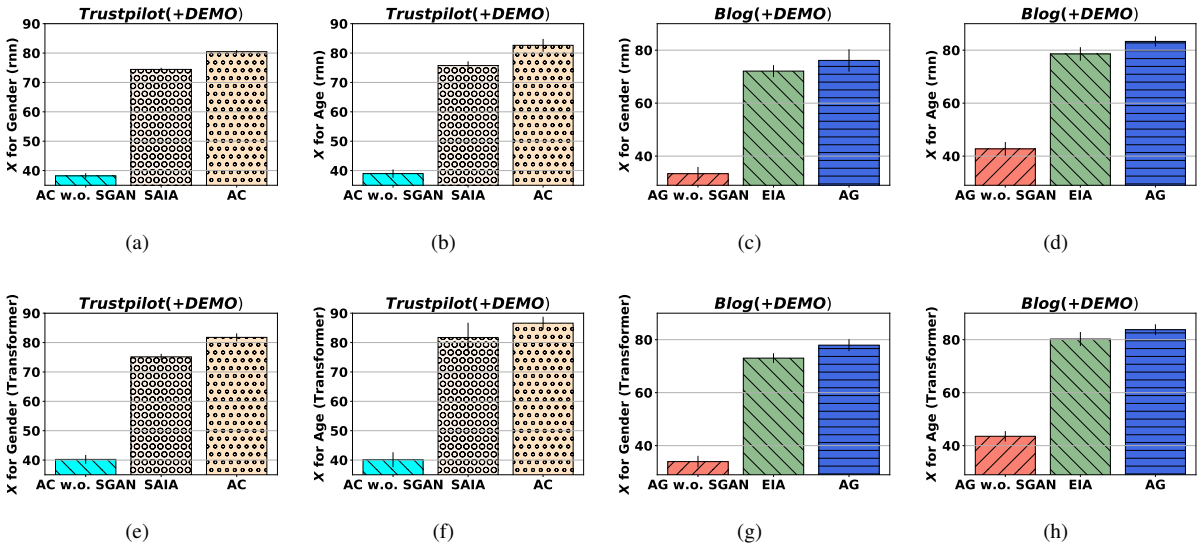
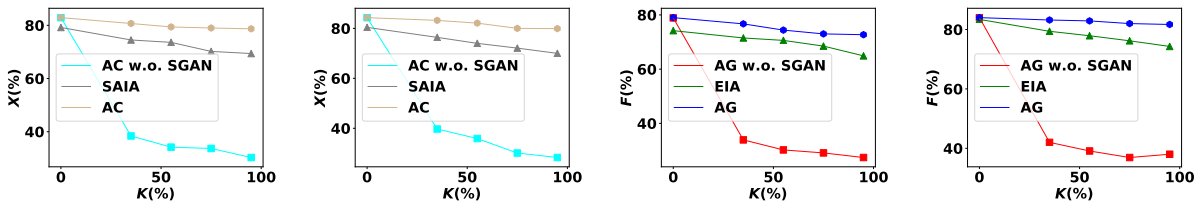


Figure 4: The performance for different sentence embeddings for both tasks in $+DEMO$ setting.



(a) Gender for Trustpilot dataset (b) Age for Trustpilot dataset (c) Gender for Blog dataset (d) Age for Blog dataset

Figure 5: The impact of K in $+DEMO$ setting.

performance in recovering the privacy such that we can claim privacy-preserving text representation is still vulnerable, **2)** how different sentence embedding methods, such as rnn and Transformer, affect the recovering ability, and **3)** how does K affects the recovering ability?

The best adversarial performance In Table 2 and Table 3, we show the performance comparisons between the best adversaries for both tasks. For all datasets, we can see that the recovering ability for Age is usually better than Gender (except the Denmark subcorpora). Besides, the best adversaries for our approach are always better than the adversary baselines. The reason is our SGAN approach is successful in distinguishing the true embeddings from the manipulated embeddings. Another observation is for $+DEMO$ setting, the adversaries achieve better performance since the privacy data are explicitly existing in the text as shown in Table 2. However, when the private information is not explicitly existing, SAIA and AC still preserve certain recovering ability as the demographic information inherently exists in the text. The results show that our method produces higher-performing models. For example, for the Trustpilot dataset, our method achieves 89.0% in terms of the accuracy for recovering private information, which is 5.9% higher than the state-of-the-art SAIA approach. For the Blog posts dataset, our method achieves 83.2% in terms of the accuracy for recovering private information, which is 3.8% higher than the EIA approach.

The performance for different sentence embeddings In Figure 4, we show the performance for different sentence embeddings for both tasks in $+DEMO$ setting, in which the first row shows the experimental results for the two tasks for the rnn embeddings and the second row shows the experimental results for the two tasks for the Transformer embeddings. Other than the recovering ability for Age is usually better than Gender (possible due to the age is more sensitive than gender for different writing styles), we also find that the transformer models achieve higher recovering ability for the private information, because of the fact that the Transformer models have more capacity in terms of rich meaning. Therefore, they are more capable of leaking private information as well.

The impact of K In this paragraph, we investigate the impact of K, as K denotes the proportion of the manipulated data that takes part in all the examples from the dataset. It's not hard to infer that as K increases, the recovering performance decreases since the lacking of the true embeddings to sufficiently train $\boldsymbol{\theta_a}$ or $\boldsymbol{\theta'_a}$. As shown in Figure 5, as K increases, the recovering performances for the three methods decrease differently. For AC w.o. SGAN and AG w.o. SGAN approaches, when $K = 0$, the recovering performances are similar to AC and AG. However, when the number of manipulated data increases, the performances for these two methods decrease rapidly. SAIA and EIA approaches are more resilient while comparing with AC w.o. SGAN and AG w.o. SGAN. However, when K increases, the performances decrease almost linearly. For our proposed AC and AG approaches, the adversary is trained by a combinatorial loss that leverages both the supervised auxiliary data and the unsupervised training data. The model doesn't need too many true embedding for training. Thus, our proposed AC and AG are the most resilient methods.

5. Discussion

Implications Recently, with an enormous amount of training data available, NLP is able to reach or even outperform human-level accuracy on many challenging tasks such as text classification [Aggarwal and Zhai, 2012], machine translation [Koehn, 2009], and speech recognition [Jelinek, 1997]. Such tremendous success leads the explosion of NLP models deployed in many Internet services and applications that people use on a daily basis, including inferring disease types from electronic health records [Gkoulalas-Divanis et al., 2014], recommending movies on Netflix [Gomez-Uribe and Hunt, 2015] or videos on YouTube [Covington et al., 2016], assisting email writing in Gmail [Wu, 2018], etc. Producing such a text classification/generative model involves a pipeline starting from collecting the user's data and training to evaluation and deployment on a server as detailed in Section 3.1. In other words, the individuals regularly create online content and organizations host online content for such systems. Usually, the quality of an ML model is measured by its predictive power on future data, and the accuracy of prediction is often the only metrics for deciding deploying ML models in production. However, the models in these services are often trained on large-scale sensitive personal data and are also making important personalized decisions for the users. The personalized nature and the decision-making role of the models provide a strong incentive for malicious adversaries to infer such models for attacking purposes (for example, inferring about the sensitive training data). It is thus crucial to understand the potential threats to such systems. As for the theoretical implications, since learning representations that exhibit invariance to

arbitrary nuisance factors yet remain useful for other tasks is challenging, learning representations of data that are invariant to changes in specified factors could be useful for a wide range of problems: removing potential biases in prediction problems, controlling the effects of covariates, and disentangling meaningful factors of variation. An existing approach [Xie et al., 2017] casts the trade-off between task performance and invariance in an adversarial way. However, Moyer et al. [2018] reported specific failure modes of adversarial training. Therefore, investigating why these methods might fail remains an open problem.

Limitations and future works Recently, a line of privacy-preserving text representation works encode the text inputs by $\Phi(\mathbf{x})$ in order to preserve the rich semantic meaning while not compromising too much utility for various downstream tasks. For example, Song and Raghunathan [2020] introduced embedding inversion attacks and attribute inference attacks. This method assumes the adversary can query Φ. This method successfully mimics the model Φ by querying a small set of auxiliary data $\mathcal{D}_a$. However, in most cases, the auxiliary data is not ideally drawn from the same underlying process from the training data. In our work, we investigate the threats for the embedding in a more realistic setting, where the server only manipulate a proportion of the training data $\mathcal{D}$ via $\Phi(\mathbf{x})$ and manipulating this set of data is sufficient for preventing the adversary from privacy recovering. The data are not manipulated (the true embeddings) remain the typical sentence embeddings. Our method investigates the threats for the embedding in a more realistic setting, where Φ is not allowed to be queried by the attacker. In our method, the server only manipulates a proportion of the training data via $\Phi(\mathbf{x})$ and manipulating this set of data is sufficient for perturbing the training of the adversary. However, our method is not applicable when $K = 1$. Thus, our future work includes generalizing our approach to the $K = 1$ case when Φ is not accessible by the adversary. Since Moyer et al. [2018] reported specific failure modes of adversarial training, another future direction of our work lies in investigating the training heuristics or other techniques that can improve the adversary's performance.

6. Conclusion

Although many recent privacy-preserving representation learning methods investigate how to protect user privacy while ensuring the utility, e.g., the accuracy of the published data for the downstream tasks, we demonstrate the new threats by researching the private information leakage for two different sentence embeddings. In particular, we propose two classes of attacks, i.e. adversarial classification (AC), and adversarial generation (AG), to study information that might be leaked by embeddings. In particular, AC shows the embeddings may reveal sensitive attributes letting alone if it explicitly exists in the input text, and AG shows the embedding vectors can be partially recovered from the input data via the generation models. Our method produces higher-performing and more resilient adversary models than the two adversary baselines. For example, our method achieves 89.0% in terms of the accuracy for recovering private information, which is 5.9% higher than the state-of-the-art SAIA approach.

Acknowledgement

The third co-author is partially supported by US National Science Foundation through the grant award OIA 1946391.

References

Charu C Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer, 2012.

Ghazaleh Beigi, Kai Shu, Ruocheng Guo, Suhang Wang, and Huan Liu. I am not what i write: Privacy preserving text representation learning. *arXiv preprint arXiv:1907.03189*, 2019.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Maximin Coavoux, Shashi Narayan, and Shay B Cohen. Privacy-preserving neural representations of text. *arXiv preprint arXiv:1808.09408*, 2018.

Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1): 2096–2030, 2016.

Aris Gkoulalas-Divanis, Grigorios Loukides, and Jimeng Sun. Publishing data from electronic health records while preserving privacy: A survey of algorithms. *Journal of biomedical informatics*, 50: 4–19, 2014.

Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

Dirk Hovy, Anders Johannsen, and Anders Søgaard. User review sites as a resource for large-scale sociolinguistic studies. In *Proceedings of the 24th international conference on World Wide Web*, pages 452–461, 2015.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. *arXiv preprint arXiv:1506.02025*, 2015.

Frederick Jelinek. *Statistical methods for speech recognition*. MIT press, 1997.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Philipp Koehn. *Statistical machine translation*. Cambridge University Press, 2009.

Yitong Li, Timothy Baldwin, and Trevor Cohn. Towards robust and privacy-preserving text representations. *arXiv preprint arXiv:1805.06093*, 2018.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

Daniel Moyer, Shuyang Gao, Rob Brekelmans, Greg Ver Steeg, and Aram Galstyan. Invariant representations without adversarial training. *arXiv preprint arXiv:1805.09458*, 2018.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, et al. Dynet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980*, 2017.

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

Daniel Preoţiuc-Pietro, Vasileios Lampos, and Nikolaos Aletras. An analysis of the user occupational class through twitter content. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1754–1764, 2015.

Sara Rosenthal and Kathleen McKeown. Age prediction in blogs: A study of style, content, and online behavior in pre-and post-social media generations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 763–772, 2011.

Jonathan Schler, Moshe Koppel, Shlomo Argamon, and James W Pennebaker. Effects of age and gender on blogging. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 6, pages 199–205, 2006.

Congzheng Song and Ananth Raghunathan. Information leakage in embedding models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 377–390, 2020.

Congzheng Song and Vitaly Shmatikov. Overlearning reveals sensitive attributes. *arXiv preprint arXiv:1905.11742*, 2019.

Yonghui Wu. Smart compose: Using neural networks to help write emails. *Google AI Blog*, 2018.

Qizhe Xie, Zihang Dai, Yulun Du, Eduard Hovy, and Graham Neubig. Controllable invariance through adversarial feature learning. *arXiv preprint arXiv:1705.11122*, 2017.