EvaLDA: Efficient Evasion Attacks Towards Latent Dirichlet Allocation

Qi Zhou,[1*] **Haipeng Chen,** [2*†] **Yitao Zheng,** [1] **Zhen Wang** [1†]
[1] School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China
[2] Center for Research on Computation and Society & Department of Computer Science,
Harvard University, Cambridge 02138, MA, USA
zhouqi@hdu.edu.cn, hpchen@seas.harvard.edu, zhengyitao@hdu.edu.cn, wangzhen@hdu.edu.cn

Abstract

As one of the most powerful topic models, Latent Dirichlet Allocation (LDA) has been used in a vast range of tasks, including document understanding, information retrieval and peer-reviewer assignment. Despite its tremendous popularity, the security of LDA has rarely been studied. This poses severe risks to security-critical tasks such as sentiment analysis and peer-reviewer assignment that are based on LDA. In this paper, we are interested in knowing whether LDA models are vulnerable to adversarial perturbations of benign document examples during inference time. We formalize the evasion attack to LDA models as an optimization problem and prove it to be *NP-hard*. We then propose a novel and efficient algorithm, EvaLDA to solve it. We show the effectiveness of EvaLDA via extensive empirical evaluations. For instance, in the NIPS dataset, EvaLDA can averagely promote the rank of a target topic from 10 to around 7 by only replacing 1% of the words with similar words in a victim document. Our work provides significant insights into the power and limitations of evasion attacks to LDA models.

Introduction

Latent Dirichlet Allocation (LDA) is one of the most powerful topic models. Due to its superiority in discovering the latent topics of documents, it has been the underlying technique in a vast range of tasks, including scientific publication understanding (Griffiths and Steyvers 2004; Talley et al. 2011), information retrieval (Harvey, Crestani, and Carman 2013), and peer-reviewer assignment (Charlin and Zemel 2013; Liu, Suel, and Memon 2014). Despite the great successes, the security of LDA-based systems has rarely been investigated. Security-critical tasks based on LDA, such as sentiment analysis and peer-reviewer assignment, are therefore at high risk of being manipulated by carefully designed adversarial attacks. For instance, by promoting the rank of a target topic for an article, it may end up in hands of colluded reviewers who are experts in the target topic. Recent studies show that machine learning (ML) methods are highly vulnerable to adversarial samples with simple and yet evasive perturbations to the benign input data (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2015). In this paper, we are interested in knowing – are LDA models vulnerable to crafted adversarial samples?

There are in general two types of adversarial attacks to ML models, namely poisoning attacks and evasion attacks (Barreno et al. 2010). Poisoning attacks aim to mislead a ML model by manipulating the training data, while evasion attacks craft malicious samples to mislead a trained ML model during test time. While there has been pioneer work (Mei and Zhu 2015) studying poisoning attacks to LDA models, no efforts have been made on investigating evasion attacks to LDA models. We present the first study of this kind. More specifically, we study evasion attacks to LDA models that are trained/inferred using the predominant Monte Carlo simulation based method called Collapsed Gibbs Sampling (CGS).[1]

Though evasion attacks of general ML models have been massively studied (Nguyen, Yosinski, and Clune 2015; Carlini and Wagner 2017; Xiao et al. 2018; Ling et al. 2019), the unique characteristics of LDA models poses several new challenges on performing evasion attacks to them. First, the input of an LDA model is a document, and is essentially textual data. Different from evasion attacks in domains like images (Nguyen, Yosinski, and Clune 2015; Carlini and Wagner 2017), textual data are intrinsically discrete, thus incurring higher difficulties to find optimal perturbations to benign samples. Second, because the input of an LDA model is an entire document, the strategy space of crafting a malicious document is huge, especially with a large document. Last and most importantly, unlike general ML models which usually infer the output via a simple forward pass (e.g., in neural networks and decision trees) from the input, the inference procedure of CGS-based LDA models involves a much more complicated Monte Carlo simulation process. This makes the gradient-based style attacks infeasible and trial-and-error style attacks extremely inefficient. [2]

In recent years, studies on evasion attacks to ML models for NLP tasks have drawn great attention from both the

[*]These authors contributed equally to this work.
[†]Corresponding authors.

[1]Another popular training/inference method for LDA models is the Variational Inference (VI) method. The two methods are distinct in the training/inference procedure. We focus on CGS based LDA models and leave VI based LDA models for future work.

[2]Note that the inference procedure for VI based LDA models is essentially an optimization. Therefore evasion attack to VI based LDA models is equally, if not more challenging.

ML and NLP communities (Papernot et al. 2016b; Ren et al. 2019; Jin et al. 2020; Behjati et al. 2019). However, most of them focus on sentence level adversarial samples, where the strategy space is much smaller than document level inputs of LDA models. More importantly, as discussed above, a key difference of LDA and general ML models is the computationally expensive inference procedure of a new input. Due to these reasons, previous works on evasion attacks to ML based NLP models cannot be applied to our problem.

We present the first study on evasion attacks to LDA. Note that our goal is not to attack the LDA models, but to provide better insights of the power and limitations of such attacks, so that appropriate defense mechanisms can be designed to prevent the attacks. We first formulate the optimal adversarial attack to LDA models as a combinatorial optimization problem. We prove that the formulated problem is *NP-hard* by reducing the combinatorial optimization problem with cardinality constraints (COPCC) (Bruglieri et al. 2006) to it.

To efficiently solve the formulated optimization problem, we propose a novel algorithm, efficient Evasion attacks towards LDA models (EvaLDA), with two key novelties. First, to handle the high computational cost of the inference procedure and thus the difficulty of estimating an adversarial perturbation's marginal contribution to the attack objective, we derive an efficient analytical estimate. The analytical estimate builds upon a carefully designed surrogate procedure for the CGS-based LDA inference. Second, due to the underlying combinatorial nature of the problem and document-level large attack strategy space, we design an efficient greedy algorithm in EvaLDA to solve the optimization problem.

Our contributions i) We are the first to study evasion attacks to LDA models. The study has significant practical impact as it provides early warning on the security vulnerability of LDA models to users and service providers. The derivation of the attack strategies provides insights into designing defense strategies against such attacks. ii) We formulate the evasion attack to LDA models as a combinatorial optimization problem and prove its *NP-hardness*. iii) We propose EvaLDA, a novel evasion attack algorithm to LDA models with key technical novelties in handling the various new challenges in the new problem. iv) We conduct extensive empirical evaluations that prove the effectiveness of EvaLDA on two distinct datasets (i.e., NIPS and AP[3]) and a large variety of problem settings. For instance, results on the NIPS dataset show that, by replacing only 1% of the words to similar words in a victim document, EvaLDA can averagely promote the rank of a target topic from 10 to around 7. We also show the effectiveness of EvaLDA via a case study. The code of this paper can be found at https://github.com/tools-only/Evasion-Attack-against-LDA-Model.

Related Work

Our work lies in the thread of research that studies evasion attacks to ML models for NLP tasks. Similar to most prior works, we aim at two goals in evasion attacks to textual data: to deteriorate the performance (e.g., accuracy) of the victim model and to keep the adversarial perturbations evasive.

[3]Links to the datasets are in the experiment section.

Gradient-based attack. Papernot et al. (2016b) make an early attempt at adversarial attacks to recurrent neural networks (RNNs) based NLP models. They calculate words' marginal contributions of classification using Fast Gradient Signed Method FGSM (Goodfellow, Shlens, and Szegedy 2015) and Forward Derivative (Papernot et al. 2016a). FGSM is also used in Samanta and Mehta (2018). They consider replacement, insertion and deletion operations to generate adversarial samples. In finding word replacements, they build a candidate replacement pool using synonyms, typos and genre specific keywords. Liang et al. (2018) calculate the character-level marginal contribution of text classification from cost gradient, and treat a word as multiple characters. This type of attacks are called "white-box" attacks as computing gradients requires knowledge about the victim model.

Black-box attack. Opposed to white-box attacks which require full knowledge about the victim model or its gradient information, black-box attacks only require knowledge about the outputs of the victim model via queries. Ren et al. (2019) study black-box attacks on text classification models. They calculate words' marginal contribution of text classification by masking out the underlying words. They also propose a scheme called probability weighted word saliency (PWWS), which considers lexical, grammatical, and semantic constraints in generating adversarial samples. Similar word selection method is used in (Jin et al. 2020), where adversarial attacks to BERT model (Devlin et al. 2019) are studied. They use word embeddings to extract synonyms of the selected words for replacement, subject to post-processing which ensures the semantics and syntax of the adversarial sentence. Alzantot et al. (2018) propose an adversarial text generation approach based on genetic algorithm, while Zhang et al. (2019) use a Markov chain Monte Carlo sampling approach.

Attack from embedding space. Compared with the above methods which work directly in discrete word or character space, Miyato, Dai, and Goodfellow (2017) propose to generate adversarial texts from the continuous embedding space. Similarly in (Behjati et al. 2019), adversarial perturbation is performed in the embedding space using loss gradient maximization. The obtained optimal perturbation is then projected back into the vocabulary space to find a feasible replacement. Sato et al. (2018) improve over (Miyato, Dai, and Goodfellow 2017) by interpretable perturbation approach which restricts perturbation to be existing points in the embedding space.

Poisoning attack to LDA. Most of the above studies on security aspects of NLP models focus on deep ML models such as RNNs (Mikolov et al. 2010) and Transformers (Vaswani et al. 2017), while little attention has been paid to the security of LDA models. A more related work is Mei and Zhu (2015), which also studies adversarial attack to LDA models. However, it is notably different from our work. First, they study poisoning attack, where the entire training corpus are target documents, whereas we study evasion attack. From a practical point of view, poisoning attacks are more difficult than evasion attacks, as it requires edit access to the entire training corpus. Second, they focus on VI (Blei, Ng, and Jordan 2003) based parameter estimation, while we study CGS (Griffiths and Steyvers 2004) based parameter estimation.

Preliminaries

As a reminder, LDA is a generative model consisting of K latent topics. It assumes the following generation process of a document m with word length N: i) Draw a "document-topic" distribution θ_m from a Dirichlet prior $\theta_m \sim \mathrm{Dir}(\alpha)$. ii) For each topic z_k in the K topics, draw a "topic-word" distribution φ_k from another Dirichlet prior: $\varphi_k \sim \mathrm{Dir}(\beta)$. Here φ_k specifies the probability distribution that the v-th word in a vocabulary $\mathcal{V}^{vocab}$ belongs to topic z_k. α and β are hyperparameters implying prior knowledge about the shape of the Dirichlet distributions. iii) For each of the N positions of the document, first sample a topic $k \sim \theta_m$, and then sample a word $w \sim \varphi_k$.

Given a set of M training documents $\mathbf{w} = \langle \mathbf{w}_m \rangle, m = 1, \dots, M$, the training process works by finding the parameter values $\theta = \langle \theta_m \rangle, m = 1, \dots, M$ and $\varphi = \langle \varphi_k \rangle, k = 1, \dots, K$ that maximizes the posterior $P(\mathbf{z}|\mathbf{w}; \theta, \varphi)$. $\mathbf{z} = \langle z_k \rangle, k = 1, \dots, K$ is a vector of latent topics. Since calculating this posterior is computational intractable, two common approximations methods are used, namely variational inference (VI) (Blei, Ng, and Jordan 2003) and collapsed Gibbs sampling (CGS) (Griffiths and Steyvers 2004). In this paper, we focus on attacks to the CGS-based LDA models.

CGS-based inference. Different from training time where both the document-topic distribution θ and topic-word distribution φ are updated, during inference time, topic-word distribution φ is usually fixed (Heinrich 2005). This is because one input sample during inference has very limited effect to φ compared with the entire training corpus.

CGS-based inference is essentially a Markov Chain Monte Carlo (MCMC) method. Given a document $\mathbf{w}_m = \langle w_{mi} \rangle, i = 1, \dots, N_m$, it works iteratively. In each iteration, it goes through each word w_{mi} in the document $\mathbf{w}_m$. At word w_{mi}, it first samples the topic z_k to be allocated at this position according to the following *full conditional distribution* (Griffiths and Steyvers 2004):

$$p\left(z^i = z_k | \mathbf{z}^{-i}, \mathbf{w}_m\right) \propto \varphi_{ki} \frac{N_{mk} + \alpha}{\sum_{k'=1}^{K} N_{mk'} + K\alpha} \tag{1}$$

where $\mathbf{z}^{-i}$ is the topic allocations of all the other words except w_{mi}. N_{mk} is the count of topic z_k being sampled in the document before word w_{mi}. After sufficient iterations (the "burn-in" period), the document-topic distribution θ_m is calculated:

$$\theta_{mk} = \frac{N_{mk} + \alpha}{\sum_{k'=1}^{K} N_{mk'} + K\alpha}, \ \forall k = 1, \dots, K \tag{2}$$

We refer to (Griffiths and Steyvers 2004; Heinrich 2005) for a detailed description for the training and inference procedures of LDA models using CGS.

Problem Formulation

Recall that in inference time, LDA maps an unseen test document $\mathbf{w}_m$ into a document-topic distribution θ_m. An evasion attack to LDA is to make perturbations to the victim document $\mathbf{w}^{vic}$ and generate an adversarial sample $\mathbf{w}^{adv}$, so that the inferred document-topic distribution θ^{vic} of the victim document is changed to $\theta^{adv} \neq \theta^{vic}$. More specifically, we consider the following adversarial word replacement attack.[4]

Definition 1. *The adversarial word replacement attack on LDA (**Attack-LDA**) problem aims to replace a subset of words $\mathcal{W}$ by a new set of words $\mathcal{W}'$, so that the victim document $\mathbf{w}^{vic}$ is changed into an adversarial document $\mathbf{w}^{adv}$, and a certain attack objective $Q(\mathcal{W}, \mathcal{W}')$ is maximized.*

Here we refer to $(\mathcal{W}, \mathcal{W}')$ as the *attack strategy*. A word $w \in \mathcal{W}$ is called a *target* word, and a word $w' \in \mathcal{W}'$ is called *replacement* word. Note that the specific form of the attack objective $Q(\mathcal{W}, \mathcal{W}')$ is dependent on the goal of attack. For instance, in a *rank promotion attack*, the goal is to maximize the increase of probability for a target topic z_k, i.e.,

$$Q(\mathcal{W}, \mathcal{W}') = \theta_k^{adv} - \theta_k^{vic} \tag{3}$$

In an *rank demotion attack*, the goal is to maximize the decrease of probability for the target topic z_k, i.e.,

$$Q(\mathcal{W}, \mathcal{W}') = \theta_k^{vic} - \theta_k^{adv} \tag{4}$$

Attack budget. As introduced in the Related Work Section, another key aspect of an evasion attack is to make minimal changes to the victim document, so as to make the attack *evasive*. This introduces a budget constraint of the form $D_w(\mathbf{w}^{vic}, \mathbf{w}^{adv}) \leq \delta$, where δ is a threshold, and $D_w(\mathbf{w}^{vic}, \mathbf{w}^{adv})$ denotes the distance of the two documents $\mathbf{w}^{vic}$ and $\mathbf{w}^{adv}$.

In practical implementation, we break it down into two types of constraints. One type of constraint specifies that the distance $D(w, w')$ of the target word w and the replacement word w' cannot exceed a threshold σ: $D(w, w') \leq \sigma, \ \forall \ (w, w') \in (\mathcal{W}, \mathcal{W}')$. An example distance measure is the cosine distance of the two words w and w' in the vector space using word embeddings (Bojanowski et al. 2017). The other type of constraint restricts that the number of words being replaced cannot exceed a certain percentage κ of the total number of words in the victim document: $|\mathcal{W}| \leq |\mathbf{w}^{vic}| \cdot \kappa$.

Formally, we represent the *Attack-LDA* problem in Definition 1 as the following optimization problem:

$$\max_{\mathcal{W}, \mathcal{W}'} \quad Q(\mathcal{W}, \mathcal{W}') \tag{5}$$

$$\text{s.t.} \quad D(w, w') \leq \sigma, \quad \forall \ (w, w') \in (\mathcal{W}, \mathcal{W}') \tag{6}$$

$$|\mathcal{W}| \leq |\mathbf{w}^{vic}| \cdot \kappa \tag{7}$$

Theorem 1. *Given an oracle that tells the algorithm the explicit value of $Q(\mathcal{W}, \mathcal{W}')$ for an attack strategy $(\mathcal{W}, \mathcal{W}')$, the Attack-LDA problem formulated above is NP-hard.*

Proof. The key idea is to reduce the (binary) combinatorial optimization problem with cardinality constraints (COPCC, which is proven to be $\mathcal{NP}$*-hard* (Bruglieri et al. 2006)) to our defined *Attack-LDA* problem.

An *arbitrary* instance of COPCC can be expressed as:

$$\min_{x} f(x) \ \text{ s.t. } \ x \in \{0,1\}^d : ||x||_0 \leq C, \tag{8}$$

[4]We only consider word replacement operation to make the paper focused. We leave the extension to other types of operations (e.g., insertion and deletion) as future work.

where x is a d-dimensional binary indicator vector which corresponds to the selection of items. That is, $x_i = 1$ indicates the i-th item is selected and otherwise not. $||x||_0$ denotes l-0 norm of x.

We then construct a *special* instance of the *Attack-LDA* problem as follows. We first let

$$Q^*(\mathcal{W}) = \max_{\mathcal{W}'} Q(\mathcal{W}, \mathcal{W}')$$
$$\text{s.t.} \quad D(w, w') \leq \sigma, \quad \forall (w, w') \in (\mathcal{W}, \mathcal{W}').$$

The *Attack-LDA* problem is then transformed as

$$\max_{\mathcal{W}} \quad Q^*(\mathcal{W}) \;\text{ s.t. } |\mathcal{W}| \leq |\mathbf{w}^{vic}| \cdot \kappa \tag{9}$$

Because finding the optimal w' for a given w requires enumerating the vocabulary space only once, therefore finding $Q^*(\mathcal{W})$ is polynomial ($\mathcal{O}(|\mathbf{w}^{vic}| \cdot |\mathcal{V}^{vocab}|)$) in the document size $|\mathbf{w}^{vic}|$ and the vocabulary size $|\mathcal{V}^{vocab}|$.

We construct the correspondence as: $d \leftrightarrow |\mathbf{w}^{vic}|$, $x_i = 1 \leftrightarrow w_i \in \mathcal{W}$, $f(x) \leftrightarrow Q^*(\mathcal{W})$, $C \leftrightarrow |\mathbf{w}^{vic}| \cdot \kappa$. In this sense, we can easily get that if x is an "yes" instance of Eq.(8), then the corresponding $\mathcal{W}$ is an "yes" instance of Eq.(9) and vice-versa. □

EvaLDA: Evasion Attack towards LDA

We can see that a key challenge of solving Eqs.(5)-(7) is to obtain the objective value $Q(\mathcal{W}, \mathcal{W}')$ given an attack strategy $(\mathcal{W}, \mathcal{W}')$. For any type of attack objectives in Eqs. (3)-(4), the key is then to compute the document-topic distributions θ^{vic} and θ^{adv} of the victim and adversarial documents. However, this is computationally expensive for CGS-based inference procedure as it involves hundreds or thousands of simulation iterations. To handle this challenge, we design a surrogate sampling-based inference procedure, from which we derive an analytical estimate of the document-topic distribution.

Another key challenge of solving the formulated optimization problem, as shown in the *NP-hardness* of the problem in Theorem 1, is the high computational complexity that arises from its combinatorial nature. To scale up EvaLDA, we propose a greedy algorithm, where we assume independence of effects on the objective function $Q(\mathcal{W}, \mathcal{W}')$ for different target-replacement word pairs (w, w').

Efficient Estimate of Document-Topic Distribution

Surrogate inference procedure. Similar to the original CGS-based inference, the surrogate inference works iteratively, where each iteration goes over all the positions of the test document $\mathbf{w}_m$ once. The *key difference* is that the topic sampling of each position happens simultaneously in each iteration. Denote the set of unique words (i.e., vocabulary) in the test document as $\mathcal{V}$, for each unique word $v \in \mathcal{V}$, it is updated n_v times, where n_v is the number of times v appears in the test document.[5] Consequently, the calculation of topic counts happens after the simultaneous sampling step.

[5]By default, we use w to denote a *position* word in the document, and v to denote a unique word in the vocabulary. Therefore in a document, there is a one to many correspondence from v to w.

Lemma 1. *In the above designed surrogate inference procedure, when $\alpha \to 0$,[6] there exists a recursive definition of the topic distribution θ_k^t for each topic $k = 1, \ldots, K$:*

$$\theta_k^t = \frac{\theta_k^{t-1}}{N} \sum\nolimits_{v \in \mathcal{V}} n_v \varphi_{kv}, \tag{10}$$

where t is the number of iterations in the CGS procedure, and $N = |\mathbf{w}_m|$ is the number of words in the test document $\mathbf{w}_m$.

Proof. At iteration t, denote the full conditional probability of sampling a topic k for word v as p_{kv}^t, then p_{kv}^t is the same at each of the n_v sampling operations:

$$p_{kv}^t = \varphi_{kv} \cdot \frac{N_k^{t-1} + \alpha}{N^{t-1} + K\alpha}$$

Here since only the test document $\mathbf{w}_m$ is involved, we have omitted the sub-script m for clarity of notation. Because the surrogate inference procedure goes over the entire document at each iteration t, the sum of topic count always equals the total number of words N in the test document. When $\alpha \to 0$ and $K\alpha \to 1$, the above equation is re-written as:

$$p_{kv}^t = \varphi_{kv} \cdot \frac{N_k^{t-1}}{N}$$

The approximation on the denominator holds as $N \gg 1$. When sampling repeats n_v times, the *expected* count of topic k being sampled at word v is $N_{kv}^t = n_v p_{kv}^t$. Note that we omit the expectation symbol for clarify of notation. Thus, the total *expected* count of topic k for the test document is:

$$N_k^t = \sum\nolimits_{v \in \mathcal{V}} n_v p_{kv}^t$$

Combining the above two equations,

$$N_k^t = \sum\nolimits_{v \in \mathcal{V}} n_v \varphi_{kv} \frac{N_k^{t-1}}{N} = \frac{N_k^{t-1}}{N} \sum\nolimits_{v \in \mathcal{V}} n_v \varphi_{kv}$$

The second equation holds because $\frac{N_k^{t-1}}{N}$ does not depend on v. According to Eq.(2) in the main text,

$$\theta_k = \frac{N_k^t + \alpha}{N + K\alpha} \to \frac{N_k^t}{N}$$

when $\alpha \to 0$, the recursive equation in Eq.(10) holds. □

From Lemma 1, we can derive the following theorem:

Theorem 2. *When $\alpha \to 0$ and the document-topic distribution is initialized as a discrete uniform distribution, i.e., for each $k = 1, \ldots, K$, there is $\theta_k^0 = 1/K$, then*

$$\theta_k^t = \frac{1}{K} \cdot \Big(\frac{\sum_{v \in \mathcal{V}} n_v \varphi_{kv}}{N} \Big)^t, \; \forall t \geq 0 \tag{11}$$

The theorem can be easily proved by induction from $t = 0$ using Lemma 1 and is therefore omitted. In practical implementation, different t values (i.e., different levels of approximation) are tested. With the analytical estimate of document-topic distribution, we can efficiently compute any form of attack objective value $Q(w, w')$ in Eqs(3)-(4) for a target-replacement word pair (w, w').

[6]Be reminded that the hyper-parameter α of the Dirichlet distribution can be interpreted as a regularization term of the document-topic distribution θ_m based on prior knowledge. In practice, α is a very small value that is approximately equal to $1/K$ where K is the number of topics. Therefore the assumption is reasonably made.

Algorithm 1: EvaLDA

```
Input: Victim document $\mathbf{w}^{vic}$, topic word distribution $\varphi$, attack type in Eqs.(3)(4), approximation level $t$ in Eq.(11), word distance threshold $\sigma$, modification budget $\kappa$.
Output: Adversarial document $\mathbf{w}^{adv}$
1  Get feasible target word set $\mathcal{W}^f$ and vocabulary $\mathcal{V}^f$
2  Get candidate replacement $\mathcal{R}(v),\ \forall v \in \mathcal{V}^f$
3  for $w \in \mathcal{W}^f$ do
4      $Q^*(w) \leftarrow 0$
5      for $w' \in \mathcal{R}(w)$ do
6          Compute $Q(w, w')$ according to Eq.(11) and attack type
7          if $Q(w, w') > Q^*(w)$ then
8              $Q^*(w) \leftarrow Q(w, w')$
9              $u(w) \leftarrow w'$
10 $\mathcal{W}^f_{sort} \leftarrow$ Sort $\mathcal{W}^f$ by $Q^*(w)$
11 $\mathcal{W}^* \leftarrow$ first word of $\mathcal{W}^f_{sort}$
12 while $|\mathcal{W}^*| < |\mathbf{w}^{vic}| \cdot \kappa$ do
13     $\mathcal{W} \leftarrow \mathcal{W}^*$
14     $w \leftarrow$ next word of $\mathcal{W}^f_{sort}$
15     $\mathcal{W}^* \leftarrow \mathcal{W}^* \cup \{w\}$
16 $\mathbf{w}^{adv} \leftarrow$ replace words $w \in \mathcal{W}$ with $u(w)$ in $\mathbf{w}^{vic}$
17 return $\mathbf{w}^{adv}$
```

Attack Strategy Design

Even with an efficient estimate of the attack objective given an attack strategy, the optimization problem in Eqs.(5)-(7) is essentially combinatorial, which is proven to be *NP-hard*. To scale up the attack, we design a greedy algorithm which assumes the marginal contribution $Q(w, w')$ to the attack objective of different target-replacement word pairs (w, w') is independent from each other:

$$Q(\mathcal{W}, \mathcal{W}') = \sum_{w \in \mathcal{W}} Q(w, w') \tag{12}$$

With this assumption, the problem is then to find the top ranked set of target-replacement word pairs which have the highest marginal contribution to the attack objective, subject to the two constraints in Eqs.(6)-(7). Algorithm 1 shows the detailed description of EvaLDA.

Step 1: Get feasible set of target words. It starts by getting the feasible set of target words $\mathcal{W}^f$ for the victim document $\mathbf{w}^{vic}$ (Line 1). This is done by removing unimportant words such as stop-words. With this step, the attack strategy space is sufficiently reduced with little sacrifice on the effectiveness of the attack. Simultaneously, we can get a feasible vocabulary $\mathcal{V}^f$ that corresponds to $\mathcal{W}^f$.

Step 2: Get candidate replacement set (Line 2). In addition to finding the top-ranked target-replacement word pairs so as to maximally deteriorate the model's performance, another critical aspect of a successful evasion attack is to make the perturbations evasive. Therefore, it is important to find a candidate set of replacements $\mathcal{R}(v)$ for each unique word $v \in \mathcal{V}^f$ that are "close" to v. In implementation, we consider two ways of building the candidate replacement set, which corresponds to two different notions of "closeness" of words. The first way is to find the set of synonyms $\mathcal{S}_v$ for each word $v \in \mathcal{V}^f$ (e.g., using WordNet[7]). Another way is to measure word "closeness" in the word embedding space, e.g. (Bojanowski et al. 2017). More specifically, we use the cosine distance $1 - \cos(v, v')$ as the distance measure of a word pair (v, v') in the embedding space and add words whose cosine distance w.r.t. the target word is smaller than a threshold σ. This is done similarly as in previous works, e.g. (Devlin et al. 2019; Abdibayev et al. 2021).

Step 3: Sort target words $w \in \mathcal{W}^f$ by their marginal contributions to the objective (Lines 3-10). This step contains two loops. The outer loop iterates over each target word w in $\mathcal{W}^f$, while for each w, the inner loop iterates over all the possible replacement word w' in $\mathcal{R}(w)$. Here $\mathcal{R}(w)$ is obtained by mapping w to the unique word v in the vocabulary. The inner loop finds the best replacement word that has the largest marginal contribution $Q(w, w')$) to the objective. For each target word $w \in \mathcal{W}^f$, its best replacement word and the corresponding largest marginal contribution are respectively denoted as $u(w)$ and $Q^*(w)$, where $Q^*(w) = \max_{w' \in \mathcal{R}(w)} Q(w, w')$. After the two loops, $\mathcal{W}^f$ is sorted w.r.t. $Q^*(w)$ to obtain the top-ranked target-replacement word pairs. The sorted *list* of words is represented as $\mathcal{W}^f_{sort}$.

Step 4: Generate adversarial sample (Lines 11-16). This step generates the attack strategy and the adversarial document. It initializes $\mathcal{W}^*$ with the first word in the sorted list $\mathcal{W}^f_{sort}$. In the while loop (Lines 12-15), it checks whether the current set of target words $\mathcal{W}^*$ exceed the budget constraint in Eq.(7). It then repeatedly adds word from $\mathcal{W}^f_{sort}$ to $\mathcal{W}^*$ until the condition is violated. Note that because $\mathcal{W}$ is a backup of $\mathcal{W}^*$ latent by one step, it is the optimal target word set after the while loop. The algorithm finally replaces each word w in $\mathcal{W}$ with $u(w)$ to obtain the adversarial document.

Experimental Evaluations

We conduct empirical experiments to evaluate EvaLDA.

Datasets We evaluate EvaLDA on 2 different datasets, NIPS[8] and AP[9]. Statistics of the datasets are shown in Table 1.

Models & hyperparameters We implement LDA-CGS using the lda package[10]. The hyperparameters of the two datasets are set as follows: the topic number is 120 for NIPS dataset, and 75 for AP dataset. The training iteration is $5,000$ which is enough to converge. We set the hyperparameters α and η of the Dirichlet distribution as default values 0.1 and 0.01. Each test sample runs 500 iterations. All experiments are run in machines with Intel E5-2678 v3 and 100GB RAM. Parameters about the EvaLDA algorithm are described after the Evaluation Metrics paragraph.

Baselines Because this is the first work that studies evasion attack to LDA models, there are no prior works that are directly

[7]http://wordnet.princeton.edu/

[8]https://www.kaggle.com/benhamner/nips-papers

[9]https://github.com/Blei-Lab/lda-c/blob/master/example/ap.tgz

[10]https://github.com/lda-project/lda

Dataset	#Train docs	#Test docs	#Train words	Vocab size	Avg test doc length
NIPS	6,562	679	9,731,300	27,176	3,211
AP	1,571	675	304,357	10,473	192

Table 1: Statistics of datasets.

applicable to our task. In order to evaluate the performance of EvaLDA, we compare it with 4 reasonable heuristic baselines. **B1** selects target-replacement word pairs completely randomly. **B2** randomly selects the target words and selects replacements as the nearest neighbor in the word vector space. **B3** selects the top-k words related to the target topic while selecting replacements randomly. **B4** selects the top-k words related to the target topic and selects replacements as the nearest neighbor in the word vector space.

Evaluation metrics We use the following 4 metrics for evaluation, where the first 2 measure the *effectiveness* of attack, and the last 2 measure the *evasiveness*. i) *Change of rank (**CoR**)* is the change of rank for a topic k: CoR $= |\text{Rank}'_k - \text{Rank}_k|$ where Rank_k and Rank'_k are respectively the rank of topic k before and after the attack. ii) *Change of probability score (**CoPS**)* means the change in probability score of target topic k: CoPS $= |\theta_k - \theta_k^{adv}|$. iii) *Average word distance (**AWD**)* is the average distance between the target-replacement word pairs in the word vector space: AWD $= 1/|\mathcal{W}| \sum_{(w,w')\in(\mathcal{W},\mathcal{W}')}(1-\cos(w,w'))$. iv) *Average sentence semantic distance* (**ASSD**). We first use BERT (Devlin et al. 2019) to encode sentences into high dimensional vectors. We then calculate the accumulated cosine distance of all sentences that are perturbed to measure the semantic distance of the victim and adversarial documents. Denote the original sentence and its perturbed version respectively as $\mathbf{s}$ and $\mathbf{s}^{adv}$, this is defined as: ASSD $= 1/|\mathcal{S}| \sum_{\mathbf{s}\in\mathcal{S}}(1-\cos(\mathbf{s},\mathbf{s}^{adv}))$, where $\mathcal{S} = \{\mathbf{s} \mid \mathbf{s} \ni w \cap w \in \mathcal{W}\}$ denotes the set of sentences which contain adversarial words. $\mathbf{s} \ni w$ means sentence s contains w. ASSD measures sentence level evasiveness.

Promotion Attack

We first evaluate the performance of promotion attack on the two datasets using the above 4 metrics, under the following settings: i) Approximation level t (in Eq.(11)) ranges over $1-6$ (default 4). ii) Word substitution strategies (Step 1 in the previous section), including synonym, word-embedding and a mixture of the two (default mixture). iii) Perturbation threshold κ (in Eq.(7)), ranges over $[0.5\%, 1\%, 2\%, 3\%]$ (default $\kappa = 1\%$). (iv) Original rank of target topic, ranges over $[5, 10, 15, 20]$ (default 10). When evaluating one parameter, we fix the other parameters with default values. We randomly choose 50 test document samples as victim samples. For all settings, we set word distance threshold $\sigma = 0.6$. For ASSD, we treat the pre- and proceeding 5 words combined of the current target word as a sentence.For the AP dataset, the different experiment settings for promotion attack are the same as before:i) Approximation level t (default 4); ii) Word substituion strategies (default mix); iii) Perturbation threshold κ, which ranges over $[1\%, 2\%, 3\%, 4\%]$ (default $\kappa = 2\%$). (iv) Original rank of target topic, which ranges over $[4, 6, 8, 10]$ (default 6). As shown in Table 1, documents in the AP data are much shorter (192 on average) than the NIPS dataset (3211 on average), using a perturbation threshold of 1% means for some documents whose length is less than 200, only one word can be perturbed. This makes it too challenging for the attack algorithms including EvaLDA. Therefore we set larger default perturbation thresholds for the experiments on the AP dataset. Similarly, because the documents are much shorter in the AP dataset, the number of topics with non-zero probability scores are also much smaller than that in the NIPS dataset. Therefore, the original rank of the target topic is set higher and denser than that in the NIPS dataset. We still randomly choose 50 samples and set word distance threshold $\sigma = 0.6$.

Approximation level Figure 1 shows the results with different t values from 1 to 6 on the *NIPS* dataset. We can see that by replacing only 1% of the words, EvaLDA significantly outperforms the baselines in promoting the rank and probability score of the target topic. In particular, the best CoR result is obtained when $t = 4$, where EvaLDA promotes the original rank from 10 to around 7 on average. We also notice that the performance of EvaLDA in all metrics is not sensitive to t values. This indicates the *robustness* of EvaLDA. Meanwhile, the AWD and ASSD values are very small (approximately 0.3 for AWD and 0.02 for ASSD). This indicates that the average perturbation per word is reasonable and the average perturbation per sentence is almost negligible. Note that **B4** achieves the best performance in AWD and ASSD. This is expected as **B4** always selects replacements as the nearest neighbor of the target word in the vector space. However, the effectiveness performance of **B4** is far worse than EvaLDA. Figure 2 shows the results for the *AP* dataset. For all the metrics, similar patterns are observed compared with experiments on the NIPS dataset. Note that we set the perturbation threshold as 2%, original rank 6 and mixed word substitution strategy in this setting. Based on this set of experiments, we choose level 4 for the rest of the experiments on both datasets.

	CoR	CoPS	AWD	ASSD
EM	2.60±0.45	0.017±0.003	0.334±0.004	0.021±0.001
Syno	0.54±0.31	0.003±0.002	**0.227±0.009**	**0.013±0.001**
Mix	**2.60±0.43**	**0.017±0.003**	0.332±0.003	0.021±0.001

Table 2: Evaluate word substitution strategies on NIPS dataset

	CoR	CoPS	AWD	ASSD
EM	0.89±0.26	0.023±0.007	0.315±0.018	0.014±0.002
Syno	0.43±0.25	0.012±0.008	**0.260±0.018**	**0.010±0.001**
Mix	**0.93±0.27**	**0.023±0.008**	0.315±0.018	0.014±0.002

Table 3: Evaluate word substitution strategies on AP dataset

Word substitution strategy Table 2 shows the result on the

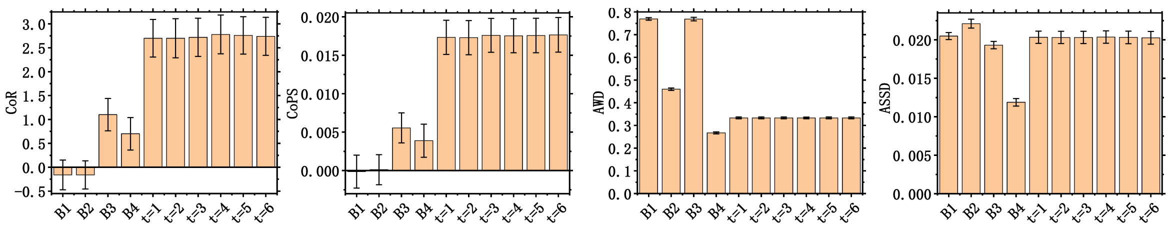


Figure 1: Promotion attack with varying approximate levels t, on the NIPS dataset, showing 95% confidence interval.

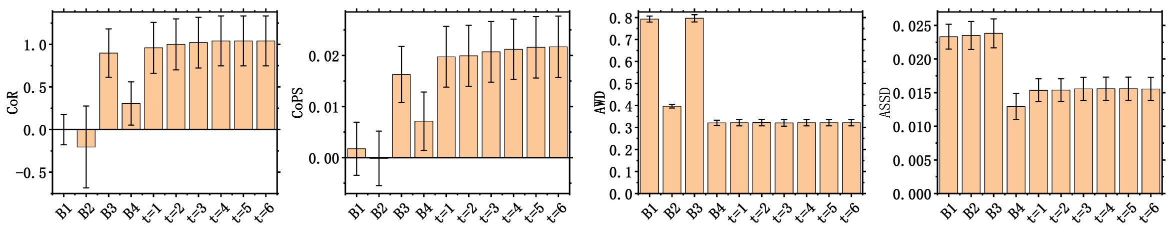


Figure 2: Promotion attack with varying approximate levels, on the AP dataset, showing 95% confidence interval.

NIPS dataset when varying word substitution strategies. Note that the performances of the baselines are not dependent on the word substitute strategies, and therefore their performances are the same as in Figure 1 and we do not repeat them here. We can see that word embedding strategy (EM) achieves much better CoR and CoPS results than word synonyms strategy (Syno), while Syno achieves better AWD and ASSD results. This is intuitive because the solution space of Syno is significantly smaller than EM, as words usually have few synonyms. The mixture of the two (Mix), achieves slightly better results in all metrics compared with EM. This indicates that Syno, while performing worst in terms of effectiveness by its own, can be a good complementary to EM. Table 3 shows the result on the *AP* dataset. Similarly, EM has better performance than Syno. A mix of the two has the best performance in CoR and CoPS, despite that AWD and ASSD of mix are almost the same as EM.

Perturbation threshold Figure 3 shows performance comparisons of EvaLDA with baselines under varying perturbation threshold values, on the *NIPS* dataset. We can see that in general, as the threshold increases, both CoR and CoPS increase. Similarly, EvaLDA is far superior to the baselines under all threshold values in terms of CoR and CoPS and is comparable to the baselines in AWD and ASSD. Be reminded that the best performance of AWD and ASSD of **B4** is due to the fact that it always selects the nearest neighbor of the target word in vector space. Interestingly, we see that the average sentence level evasiveness value (ASSD) of EvaLDA decreases when threshold value increases. Our conjecture for this is when threshold increases (i.e., solution space is larger), the newly found target-replacement word pairs by EvaLDA are closer. Figure 4 shows the result on the *AP* dataset. In terms of effectiveness (CoR and CoPS), EvaLDA is superior to the baseline methods under almost all threshold values. Note that **B3** also performs well under different perturbation thresholds. This is because **B3** selects a random replacement word for the target word and it is more likely to shift the target topic. However, such attacks are impractical as it is highly detectable due to the large word-level and sentence variations, which are indicated by the very large AWD and ASSD values.

Original rank of target topic Intuitively, promotion attacks on topics that are ranked lower should be more effective than topics that are ranked higher. Results in both Figures 5 and 6 follow this intuition. In addition, since EvaLDA does not consider the rank of the original topic in the optimization process, it works constantly in CoPS regardless of the original rank. Moreover, we can see that the AWD values of EvaLDA are comparable with **B1**-**B3** and slightly worse than **B4**. As shown in Figure 6, the overall results on the *AP* dataset are similar to those of the *NIPS* dataset. The lower ranked topics are easier to promote, even if topics with higher rankings can get more score promotion, its ranks are still relatively stable. As we can see, EvaLDA performs better in CoR and CoPS, and also competitive in AWD and ASSD.

Demotion Attack

As shown in Eq.(4), rank demotion attack aims at decreasing the rank of a target topic. Figure 7 shows the result of rank demotion attack with varying original rank of the target topic, on the *NIPS* dataset. We can see that, intuitively, the CoPS value decreases when the original rank is lower. However, the CoR metric does not decrease w.r.t. the original rank. This seems to be counter-intuitive, but in fact is correct because although the attack to originally higher ranked topics achieves higher CoPS values, the original probability scores of these highly ranked topics are also higher. Therefore it is difficult to change it from a high score to a very low score, rendering it difficult to demote. In terms of AWD and ASSD, EvaLDA performs even better than **B4**. This is because the solution

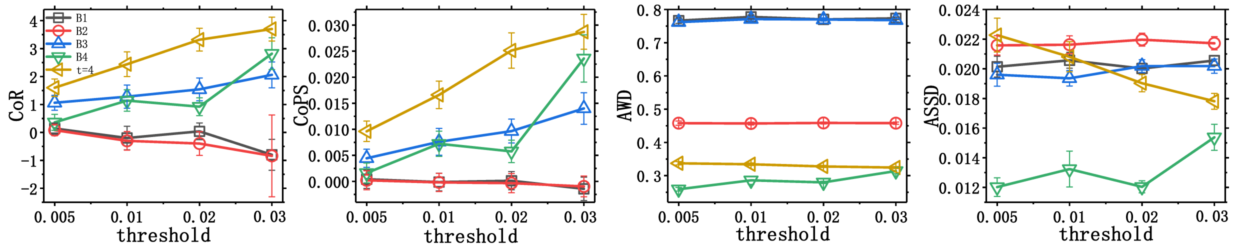


Figure 3: Promotion attack with varying perturbation threshold, on the NIPS dataset, showing 95% confidence interval.

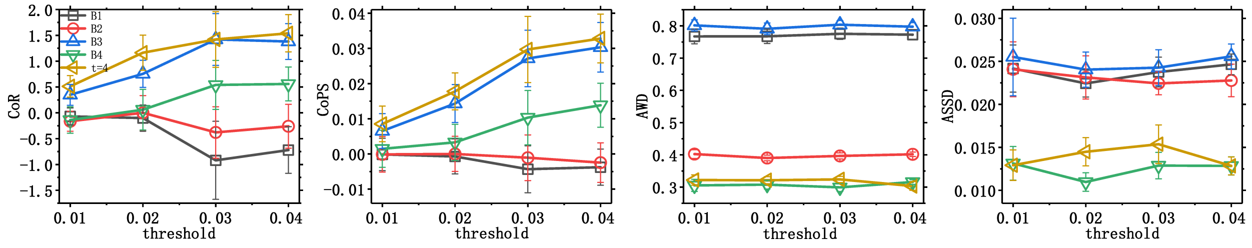


Figure 4: Promotion attack with varying perturbation threshold, on the AP dataset, showing 95% confidence interval.

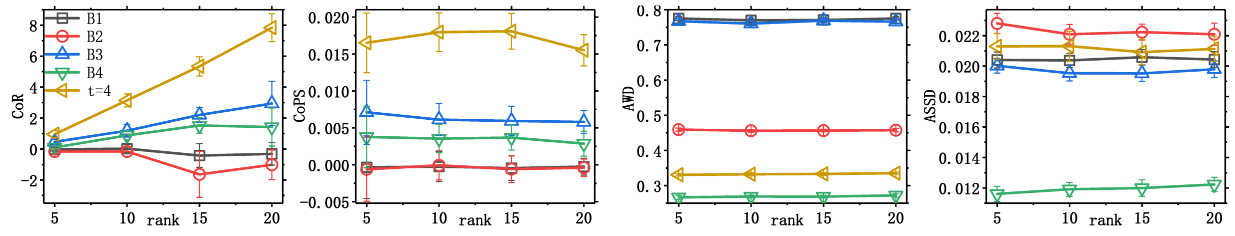


Figure 5: Promotion attack with varying original rank of the target topic, on the NIPS dataset, showing 95% confidence interval.

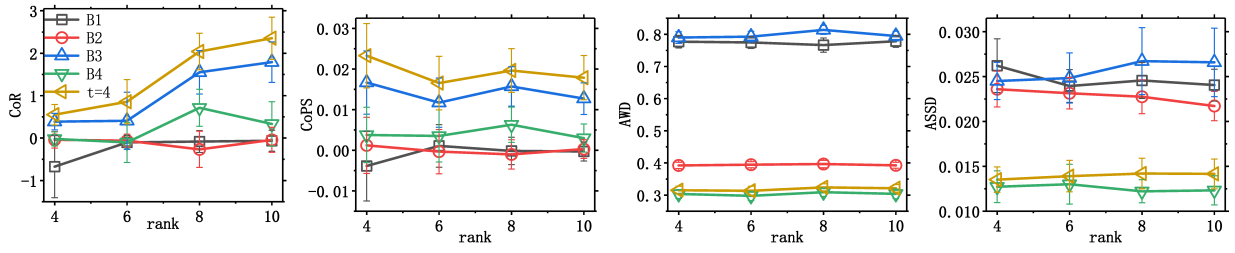


Figure 6: Promotion attack with varying original rank of the target topic, on the AP dataset, showing 95% confidence interval.

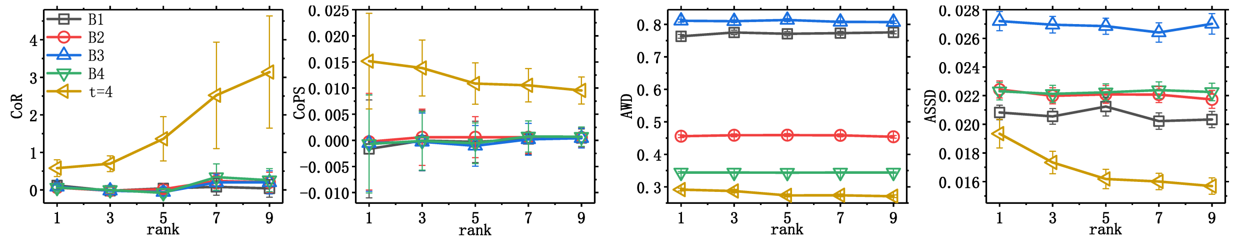


Figure 7: Demotion attack with varying original rank of target topic, on the NIPS dataset, showing 95% confidence interval.

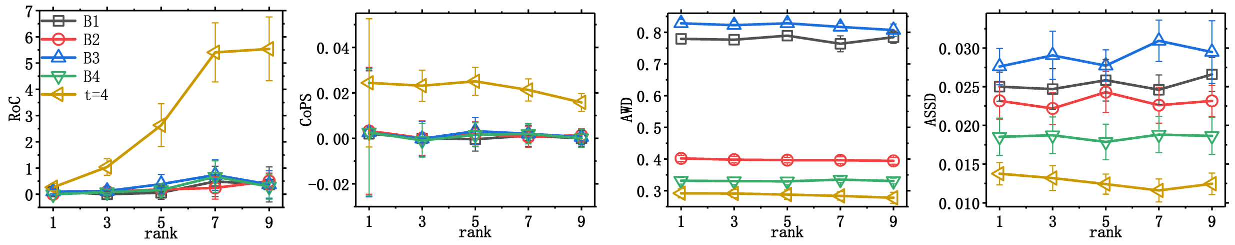
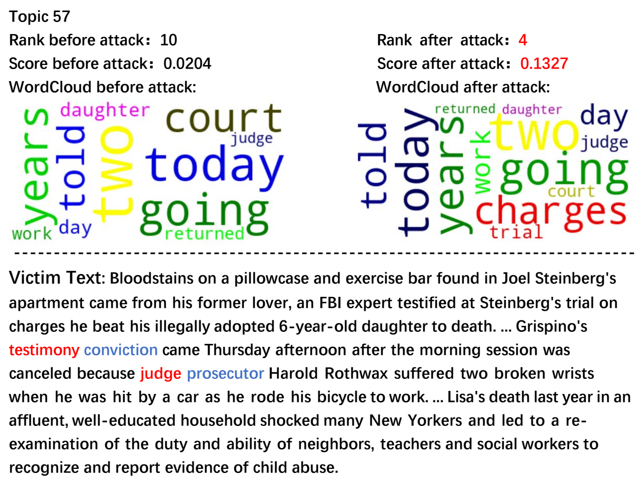


Figure 8: Demotion attack with varying original rank of target topic, on the AP dataset, showing 95% confidence interval.



Victim Text: Bloodstains on a pillowcase and exercise bar found in Joel Steinberg's apartment came from his former lover, an FBI expert testified at Steinberg's trial on charges he beat his illegally adopted 6-year-old daughter to death. ... Grispino's testimony conviction came Thursday afternoon after the morning session was canceled because judge prosecutor Harold Rothwax suffered two broken wrists when he was hit by a car as he rode his bicycle to work. ... Lisa's death last year in an affluent, well-educated household shocked many New Yorkers and led to a re-examination of the duty and ability of neighbors, teachers and social workers to recognize and report evidence of child abuse.

Figure 9: An adversarial sample generated by EvaLDA. Target and replacement words are resp. in red and blue. By replacing 2 words "testimony" and "judge" with "conviction" and "prosecutor", the rank of the topic is greatly promoted.

space is larger than that of promotion attack, so EvaLDA can achieve aggressive goals while ensuring similarity. Figure 8 shows the topic demotion attack performance on the *AP* dataset. We can see that EvaLDA beats the baselines by a very large margin for CoR and CoPS, while having the best AWD and ASSD values too, which are even smaller than **B4**.

Case Study

We now show via a concrete example how EvaLDA works. Because documents from the AP dataset are shorter and the contents are news from Associated Press, we select a victim document sample from the AP dataset. The length of the document is 111 words. We generate an adversarial sample using EvaLDA with $t = 4$ and perturbation threshold $\kappa = 0.02$ – which means we can perturb only 2 words. Figure 9 shows a piece of the sample document, where target and replacement words are respectively in red and blue. We choose topic 57 as the target topic, which ranks 10th originally. We can see that by replacing the words "testimony" and "judge" with reasonably close words "conviction" and "prosecutor", the rank of the topic is promoted to 4 and the topic probability score increases from 0.0204 to 0.1327. At the top of the text, the topic WordCloud before and after the attack also shows that the two new words "charges" and "trial" appear with evidently greater weights.

Conclusion, Limitations, and Future Work

This work is the first to study evasion attacks to LDA models. The formulated optimization problem, which is provably *NP-hard*, is solved via our proposed novel algorithm EvaLDA. EvalDA consists of an efficient estimate of the topic-word distribution via a surrogate CGS-based inference procedure and a greedy target word selection and replacement procedure. Via extensive experimental evaluations on two distinct datasets, we show that EvaLDA achieves superb performances in both topic rank promotion and demotion attacks under various settings. Despite the advantages, EvaLDA does have a few *limitations*. First, it is limited to attacks to CGS-based LDA models. It remains unclear whether it works on VI-based LDA models. A potential idea is via attack strategy transferred from a CGS-based LDA model. Second, EvaLDA assumes a white-box setting and acts more like a proof-of-concept analysis in the worst-case scenario. It is interesting to see whether attacks can be effective in a black-box setting. Another critical future work, which is also the ultimate goal of studying adversarial attacks to LDA models, is to design effective defense strategies to against such attacks.

Acknowledgements

The paper is supported by the National Natural Science Foundation of China (Grant No. 62002092). H. C. is funded by the Harvard Center for Research on Computation and Society.

Ethical Impact

We aim at providing insights into the power and limitations of adversarial attacks towards LDA models. By exposing this work, users and service providers of LDA models can be alerted the potential risks of such models. Moreover, design defense strategies towards such attacks can be designed to improve the robustness of the models. The potential downside is that malicious hackers may be aware of the vulnerability of the LDA models and make exploits of it. However, without being aware of the vulnerabilities would pose critical risk of their product and systems. Therefore, exposing such vulnerabilities of LDA models is the necessary first step.

References

Abdibayev, A.; Chen, D.; Chen, H.; Poluru, D.; and Subrahmanian, V. 2021. Using Word Embeddings to Deter Intellectual Property Theft through Automated Generation of Fake Documents. *ACM Transactions on Management Information Systems (TMIS)* 12(2): 1–22.

Alzantot, M.; Sharma, Y. S.; Elgohary, A.; Ho, B.-J.; Srivastava, M.; and Chang, K.-W. 2018. Generating Natural Language Adversarial Examples. In *EMNLP*.

Barreno, M.; Nelson, B.; Joseph, A. D.; and Tygar, J. D. 2010. The security of machine learning. *Machine Learning* 81(2): 121–148.

Behjati, M.; Moosavi-Dezfooli, S.-M.; Baghshah, M. S.; and Frossard, P. 2019. Universal adversarial attacks on text classifiers. In *ICASSP*, 7345–7349.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan): 993–1022.

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5: 135–146.

Bruglieri, M.; Ehrgott, M.; Hamacher, H. W.; and Maffioli, F. 2006. An annotated bibliography of combinatorial optimization problems with fixed cardinality constraints. *Discrete Applied Mathematics* 154(9): 1344–1357.

Carlini, N.; and Wagner, D. 2017. Towards evaluating the robustness of neural networks. In *IEEE SP*, 39–57.

Charlin, L.; and Zemel, R. 2013. The Toronto paper matching system: an automated paper-reviewer assignment system. In *Proceedings of the ICML Workshop on Peer Reviewing and Publishing Models (PEER)*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

Griffiths, T. L.; and Steyvers, M. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences* 101(suppl 1): 5228–5235.

Harvey, M.; Crestani, F.; and Carman, M. J. 2013. Building user profiles from topic models for personalised search. In *CIKM*, 2309–2314.

Heinrich, G. 2005. Parameter estimation for text analysis. Technical report, Fraunhofer IGD.

Jin, D.; Jin, Z.; Zhou, J. T.; and Szolovits, P. 2020. Is bert really robust? natural language attack on text classification and entailment. In *AAAI*.

Liang, B.; Li, H.; Su, M.; Bian, P.; Li, X.; and Shi, W. 2018. Deep text classification can be fooled. In *IJCAI*, 4208–4215.

Ling, X.; Ji, S.; Zou, J.; Wang, J.; Wu, C.; Li, B.; and Wang, T. 2019. Deepsec: A uniform platform for security analysis of deep learning model. In *IEEE SP*, 673–690.

Liu, X.; Suel, T.; and Memon, N. 2014. A robust model for paper reviewer assignment. In *ACM RecSys*, 25–32.

Mei, S.; and Zhu, X. 2015. The security of latent dirichlet allocation. In *AISTATS*, 681–689.

Mikolov, T.; Karafiát, M.; Burget, L.; Černockỳ, J.; and Khudanpur, S. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*.

Miyato, T.; Dai, A. M.; and Goodfellow, I. 2017. Adversarial training methods for semi-supervised text classification. In *ICLR*.

Nguyen, A.; Yosinski, J.; and Clune, J. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 427–436.

Papernot, N.; McDaniel, P.; Jha, S.; Fredrikson, M.; Celik, Z. B.; and Swami, A. 2016a. The limitations of deep learning in adversarial settings. In *EuroS&P*, 372–387. IEEE.

Papernot, N.; McDaniel, P.; Swami, A.; and Harang, R. 2016b. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM*, 49–54.

Ren, S.; Deng, Y.; He, K.; and Che, W. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *ACL*, 1085–1097.

Samanta, S.; and Mehta, S. 2018. Towards crafting text adversarial samples. In *ECIR*.

Sato, M.; Suzuki, J.; Shindo, H.; and Matsumoto, Y. 2018. Interpretable adversarial perturbation in input embedding space for text. In *IJCAI*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Estrach, J. B.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Talley, E. M.; Newman, D.; Mimno, D.; Herr II, B. W.; Wallach, H. M.; Burns, G. A.; Leenders, A. M.; and McCallum, A. 2011. Database of NIH grants using machine-learned categories and graphical clustering. *Nature Methods* 8(6): 443.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Neurips*, 5998–6008.

Xiao, C.; Li, B.; Zhu, J.-Y.; He, W.; Liu, M.; and Song, D. 2018. Generating adversarial examples with adversarial networks. In *IJCAI*, 3905–3911.

Zhang, H.; Zhou, H.; Miao, N.; and Li, L. 2019. Generating Fluent Adversarial Examples for Natural Languages. In *ACL*, 5564–5569.