Learning to Discriminate Perturbations for Blocking Adversarial Attacks in Text Classification

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang and Wei Wang
Computer Science Department
University of California, Los Angeles
\{yz, jyunyu, kwchang, weiwang\}@cs.ucla.edu

Abstract

Adversarial attacks against machine learning models have threatened various real-world applications such as spam filtering and sentiment analysis. In this paper, we propose a novel framework, learning to discriminate perturbations (DISP), to identify and adjust malicious perturbations, thereby blocking adversarial attacks for text classification models. To identify adversarial attacks, a perturbation discriminator validates how likely a token in the text is perturbed and provides a set of potential perturbations. For each potential perturbation, an embedding estimator learns to restore the embedding of the original word based on the context and a replacement token is chosen based on approximate kNN search. DISP can block adversarial attacks for any NLP model without modifying the model structure or training procedure. Extensive experiments on two benchmark datasets demonstrate that DISP significantly outperforms baseline methods in blocking adversarial attacks for text classification. In addition, in-depth analysis shows the robustness of DISP across different situations.

1 Introduction

Deep learning techniques (Goodfellow et al., 2016) have achieved enormous success in many fields, such as computer vision and NLP. However, complex deep learning models are often sensitive and vulnerable to a tiny modification. In other words, malicious attackers can destroy the models by adding a few inconspicuous perturbations into input data, such as masking images with unrecognizable filters and making low-key modifications for texts. Therefore, developing techniques to equip models against adversarial attacks becomes a prominent research problem.

Existing studies on adversarial attacks can be classified into two groups, generation of adversarial examples and defense against adversarial attacks (Yuan et al., 2019). In the field of NLP, most of the existing studies focus on the former. For example, Ebrahimi et al. (2017); Alzantot et al. (2018) replace a word with synonyms or similar words while Gao et al. (2018); Liang et al. (2017); Ebrahimi et al. (2017) conduct character-level manipulations to fool the models. Moreover, it is not straightforward to adapt existing approaches for blocking adversarial attacks, such as data augmentation (Krizhevsky et al., 2012; Ribeiro et al., 2018; Ren et al., 2019) and adversarial training (Goodfellow et al., 2015; Iyyer et al., 2018; Marzinotto et al., 2019; Cheng et al., 2019; Zhu et al., 2019), to NLP applications. Hence, the defense against adversarial attacks in NLP remains a challenging and unsolved problem.

Recognizing and removing the inconspicuous perturbations are the core of defense against adversarial attacks. For instance, in computer vision, denoising auto-encoders (Warde-Farley and Bengio, 2017; Gu and Rigazio, 2015) are applied to remove the noises introduced by perturbations; Prakash et al. (2018) manipulate the images to make the trained models more robust to the perturbations; Samangouei et al. (2018) apply generative adversarial networks to generate perturbation-free images. However, all of these approaches cannot straightforwardly apply to the NLP tasks for the following two reasons. First, images consist of continuous pixels while texts are discrete tokens. As a result, a token can be replaced with another semantically similar token that drops the performance, so perturbations with natural looks cannot be easily recognized compared to previous approaches that capture unusual differences between the intensities of neighboring pixels. Second, sentences consist of words with an enormous
vocabulary size, so it is intractable to enumerate all of the possible sentences. Therefore, existing defense approaches in computer vision that rely on pixel intensities cannot be directly used for the NLP tasks.

After recognizing the perturbed tokens, the naïve way to eliminate the perturbations for blocking adversarial attacks is to remove these perturbed tokens. However, removing words from sentences results in fractured sentences, causing the performance of NLP models to degrade. Therefore, it is essential to recover the removed tokens. Nevertheless, training a satisfactory language model requires myriad and diverse training data, which is often unavailable. An inaccurate language model that incoherently patches missing tokens can further worsen the prediction performance. To tackle this difficult challenge, we propose to recover the tokens from discriminated perturbations by a masked language model objective with contextualized language modeling.

In this paper, we propose Learning to Discriminate Perturbations (DISP), as a framework for blocking adversarial attacks in NLP. More specifically, we aim to defend the model against adversarial attacks without modifying the model structure and the training procedure. DISP consists of three components, perturbation discriminator, embedding estimator, and hierarchical navigable small world graphs. Given a perturbed testing data, the perturbation discriminator first identifies a set of perturbed tokens. For each perturbed token, the embedding estimator optimized with a corpus of token embeddings infers an embedding vector to represent its semantics. Finally, we conduct an efficient kNN search over a hierarchical taxonomy to translate each of the embedding vectors into appropriate token to replace the associated perturbed word. We summarize our contributions in the following.

• To the best of our knowledge, this paper is the first work for blocking adversarial attacks in NLP without retraining the model.

• We propose a novel framework, DISP, which is effective and significantly outperforms other baseline methods in defense against adversarial attacks on two benchmark datasets.

• Comprehensive experiments have been conducted to demonstrate the improvements of DISP. In addition, we will release our implementations and the datasets to provide a testbed and facilitate future research in this direction.

2 Related Work

Adversarial examples crafted by malicious attackers expose the vulnerability of deep neural networks when they are applied to down-streaming tasks, such as image recognition, speech processing, and text classifications (Wang et al., 2019; Goodfellow et al., 2015; Nguyen et al., 2015; Moosavi-Dezfooli et al., 2017).

For adversarial attacks, white-box attacks have full access to the target model while black-box attacks can only explore the models by observing the outputs with limited trials. Ebrahimi et al. (2017) propose a gradient-based white-box model to attack character-level classifiers via an atomic flip operation. Small character-level transformations, such as swap, deletion, and insertion, are applied on critical tokens identified with a scoring strategy (Gao et al., 2018) or gradient-based computation (Liang et al., 2017). Samanta and Mehta (2017); Alzantot et al. (2018) replace words with semantically and syntactically similar adversarial examples.

However, limited efforts have been done on adversarial defense in the NLP fields. Texts as discrete data are sensitive to the perturbations and cannot transplant most of the defense techniques from the image processing domain such as Gaussian denoising with autoencoders (Meng and Chen, 2017; Gu and Rigazio, 2014). Adversarial training is the prevailing counter-measure to build a robust model (Goodfellow et al., 2015; Iyyer et al., 2018; Marzinotto et al., 2019; Cheng et al., 2019; Zhu et al., 2019) by mixing adversarial examples with the original ones during training the model. However, these adversarial examples can be detected and deactivated by a genetic algorithm (Alzantot et al., 2018). This method also requires retraining, which can be time and cost consuming for large-scale models.

Spelling correction (Mays et al., 1991; Islam and Inkpen, 2009) and grammar error correction (Sakaguchi et al., 2017) are useful tools which can block editorial adversarial attacks, such as swap and insertion. However, they cannot handle cases where word-level attacks that do not cause spelling and grammar errors. In our paper, we propose a general schema to block both word-level and character-level attacks.
3 DISP for Blocking Adversarial Attacks

In this section, we first formally define the goal of adversarial defense and then introduce the proposed framework DISP, learning to discriminate perturbations, for blocking adversarial attacks.

Problem Statement. Given an NLP model \( F(X) \), where \( X = \{t_1, \ldots, t_N\} \) is the input text of \( N \) tokens while \( t_i \) indicates the \( i \)-th token. A malicious attacker can add a few inconspicuous perturbations into the input text and generate an adversarial example \( X_a \) so that \( F(X) \neq F(X_a) \) with unsatisfactory prediction performance. For example, a perturbation can be an insertion, a deletion of a character in a token, a replacement of a token with its synonym. In this paper, we aim to block adversarial attacks for general text classification models. More specifically, we seek to preserve the model performances by recovering original input text and universally improve the robustness of any text classification model.

3.1 Framework Overview

Figure 1 illustrates the overall schema of the proposed framework. DISP consists of three components, (1) a perturbation discriminator, (2) an embedding estimator, and (3) a token embedding corpus with the corresponding small world graphs \( G \).

In the training phase, DISP constructs a corpus \( D \) from the original corpus for training the perturbation discriminator so that it is capable of recognizing the perturbed tokens. The corpus of token embeddings \( C \) is then applied to train the embedding estimator to recover the removed tokens after establishing the small world graphs \( G \) of the embedding corpus. In the prediction phase, for each token in testing data, the perturbation discriminator predicts if the token is perturbed. For each potential perturbation that is potentially perturbed, the embedding estimator generates an approximate embedding vector and retrieve the token with the closest distance in the embedding space for token recovery. Finally, the recovered testing data can be applied for prediction. Note that the prediction model can be any NLP model. Moreover, DISP is a general framework for blocking adversarial attacks, so the model selection for the discriminator and estimator can also be flexible.

3.2 Perturbation Discrimination

Perturbation Discriminator. The perturbation discriminator plays an important role to classify whether a token \( t_i \) in the input \( X_a \) is perturbed based on its neighboring tokens. We adopt contextualized language modeling, such as BERT (Devlin et al., 2018), to derive \( d \)-dimension contextualized token representation \( T^D_i \) for each token \( t_i \) and then cascade it with a binary logistic regression classifier to predict if the token \( t_i \) is perturbed or not. Figure 2 illustrates the perturbation discriminator based on a contextualized word encoder. The discriminator classifies a token \( t_i \) into two classes \( \{0, 1\} \) with logistic regression based on the contextual representation \( T^D_i \) to indicate if the token is perturbed. More formally, for each token \( t_i \), the discriminator predictions \( r_i \) can then be derived as:

\[
r_i = \arg\max_c y^c_i = \arg\max_c (w_c \cdot T^D_i + b_c),
\]

where \( y^c_i \) is the logit for the class \( c \); \( w_c \) and \( b_c \) are the weights and the bias for the class \( c \). Finally, the potential perturbations \( R \) is the set of tokens with positive discriminator predictions \( R = \{t_i | r_i = 1\} \).

3.3 Efficient Token-level Recovery with Embedding Estimator

After predicting the perturbations \( R \), we need to correct these disorders to preserve the prediction performance. One of the most intuitive approaches to recover tokens with context is to exploit language models. However, language models require sufficient training data while the precision to exact tokens can be dispensable for rescuing prediction performance. Moreover, over-fitting limited training data can be harmful to the prediction quality. To resolve this problem, we assume that replacing the perturbed word with a word with similar meanings to the original word is sufficient for the downstream models to make the correct prediction. Based on the assumption, DISP first predicts the embeddings of the recovered tokens for the potential perturbations with an embedding estimator based on context tokens. The tokens can then be appropriately recovered by an efficient \( k \)-nearest neighbors (\( k \)-NN) search in the embedding space of a token embedding corpus \( C \).

Embedding Estimator. Similar to the perturbation discriminator, any regression model can be employed as an embedding estimator based on the proposed concept. Here we adopt the contextualized language modeling again as an example of the embedding estimator. For each token \( t_i \), the
contextualized token embedding can be derived as a d-dimensional contextual representation vector 
$T_i^G$ to be features for estimating appropriate 
embeddings.

Figure 3 shows the embedding estimator based 
on BERT. For each potential perturbation $t_i \in R$, 
$2w$ neighboring tokens are selected as the context 
for estimating the appropriate embedding, where 
$w$ decides the window size. More precisely, a segment 
of tokens with a window size $2w + 1$ from 
$t_{i-w}$ to $t_{i+w}$ is the input tokens for BERT, where 
$t_i$ is replaced with a [MASK] token as the perturbed 
position. Finally, for the target $t_i$, a weight matrix 
$W^G \in \mathbb{R}^{d \times k}$ projects the contextual representation 
$T_i^G$ to a $k$-dimensional estimated embedding 
e_i as follows:

$$e_i = T_i^G W^G,$$

where the dimension size $k$ is required to be consistent 
with the embedding dimension in the token embedding corpus $C$.

**Efficient Token-level Recovery.** Finally, we re-
cover the input sentence based on the predicted re-
cover embeddings from the embedding estimator. 
Specifically, the input text $X$ needs to be recov-
ered from the perturbed text $X_a$ by fixing token-
level perturbations based on its approximate embed-
ments.

Given the token embedding corpus $C$, it is 
simple to transform an embedding to a token by 
finding the nearest neighbor token in the embed-
ding space. However, a naïve kNN search query 
can take $O(kn)$ time complexity, where $n$ is the 
number of embeddings in $C$; $k$ is the embed-
ding dimension. To accelerate the search pro-
cess, we apply hierarchical navigable small world 
graphs (SWGs) (Malkov and Yashunin, 2018) for 
fast approximate kNN search. More precisely, em-

---

**Figure 1:** Schema of the proposed framework DISP.

**Figure 2:** The illustration of the perturbation discrimi-
nator in DISP.

**Figure 3:** The illustration of the embedding estimator 
in DISP with a window size $2w + 1$ for the token at the 
position $i$. 

---

4906
Algorithm 1: Efficient Token-level Recovery

**Input:** Perturbed text $X_a$, potential perturbations $R$, estimated embeddings $\{e_i\}$, small world graphs $G$, token embedding corpus $C$.

**Output:** Recovered text $X_r$.

1. $X_r = X_a$
2. for $t_i \in R$ do
   3. index = QuerySmallWorldGraph($G, e_i$);
   4. $z = C[index].token$
   5. Replace $t_i$ in $X_r$ with $z$
6. return $X_r$

Embeddings are transformed into a hierarchical set of SWGs based on the proximity between different embeddings. To conduct $k$NN searches, the property of degree distributions in SWGs significantly reduces the search space of each $k$NN query from $O(n)$ to $O(\log n)$ by navigating on the graphs, so a $k$NN query can be efficiently completed in $O(k \log n)$ time complexity. Finally, the recovered text $X_r$ can be obtained by replacing the perturbations $R$ in $X_a$ as shown in Algorithm 1.

3.4 Learning and Optimization

To learn a robust discriminator, we randomly sample adversarial examples from both character-level and word-level attacks in each training epoch. The loss function optimizes the cross-entropy between the labels and the probabilistic scores computed by the logits $y_i$ and the softmax function.

The learning process of embedding estimator is similar to masked language models. The major difference is that language models optimize the likelihood to generate the same original token while the embedding estimator minimizes the distance between the derived embedding and the original token embedding. To learn the embedding estimator, a size-$(2w + 1)$ sliding window is applied to enumerate $(2w + 1)$-gram training data for approximating embeddings with context tokens. For optimization, the embedding estimator is learned to minimize the mean square error (MSE) from the inferred embeddings to the original token embeddings.

To take advantage of hierarchical navigable SWGs for an efficient recovery, although a preprocess to construct SWGs $G$ is required, the preprocess can be fast. The established SWGs can also be serialized in advance. More precisely, the time complexity is $O(kn \log n)$ for one-time construction of reusable SWGs, where $n$ is the number of embeddings in the embedding corpus $C$.

4 Experiments

In this section, we conduct extensive experiments to evaluate the performance of DISP in improving model robustness.

4.1 Experimental Settings

**Experimental Datasets.** Experiments are conducted on two benchmark datasets: (1) Stanford Sentiment Treebank Binary (SST-2) (Socher et al., 2013) and (2) Internet Movie Database (IMDb) (Maas et al., 2011). SST-2 and IMDb are both sentiment classification datasets which involve binary labels annotating sentiment of sentences in movie reviews. Detailed statistics of two datasets are listed in Table 1.

**Attack Generation.** We consider three types of character-level attacks and two types of word-level attacks. The character-level attacks consist of insertion, deletion, and swap. Insertion and deletion attacks inject and remove a character, respectively, while a swap attack flips two adjacent characters. The word-level attacks include random and embed. A random attack randomly samples a word to replace the target word while an embed attack replaces the word with a word among the top-10 nearest words in the embedding space. The examples of each attack type are illustrated in Table 2. To obtain strong adversarial attack samples, we consider to leverage oracle to identify the perturbations that cause prediction changes. Specifically, for each test sample we construct 50 adversarial examples by perturbing the test data. We sample one example in which model predictions change after perturbing. If none of them can

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Train</th>
<th>Test</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>SST-2</td>
<td>67,349</td>
<td>1,821</td>
<td>56</td>
</tr>
<tr>
<td>IMDb</td>
<td>25,000</td>
<td>25,000</td>
<td>2,738</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Attack</td>
<td>Old-form moviemaking at its best.</td>
</tr>
<tr>
<td>Insertion</td>
<td>Old-form moviemaking at its beast.</td>
</tr>
<tr>
<td>Deletion</td>
<td>Old-form moviemaking at its best.</td>
</tr>
<tr>
<td>Swap</td>
<td>Old-form moviemaking at its bets.</td>
</tr>
<tr>
<td>Random</td>
<td>Old-form moviemaking at its aggrandize.</td>
</tr>
<tr>
<td>Embed</td>
<td>Old-form moviemaking at its way.</td>
</tr>
</tbody>
</table>
change the prediction, the sample with the least
certainty is selected.

**Base Model and Baselines.** We consider
BERT (Devlin et al., 2018) as the base model as it
achieves strong performance in these benchmarks.
To evaluate the performance of DISP, we con-
sider the following baseline methods: (1) Adver-
sarial Data Augmentation (ADA) samples adver-
sarial examples to increase the diversity of training
data; (2) Adversarial Training (AT) samples differ-
ten adversarial examples in each training epoch;
(3) Spelling Correction (SC) is used as a base-
line for discriminating perturbations and blocking
character-level attacks. Note that ADA and AT re-
quire to re-train BERT with the augmented train-
ing data, while DISP and SC modify the input text
and then exploit the original model for prediction.
SC is also the only baseline for evaluating discrim-
inator performance. In addition, we also try to en-
ssemble DISP and SC (DISP+SC) by conducting
DISP on the spelling corrected input.

**Evaluation Metrics.** We evaluate the perfor-
mance of the perturbation discriminator by preci-
sion, recall and F1 scores, and evaluate the overall
disruptive performance by classification accuracy
that the models recover.

**Implementation Details.** The model is imple-
mented in PyTorch (Paszke et al., 2017). We set
the initial learning and dropout parameter to be
$2 \times 10^{-5}$ and 0.1. We use crawl-300d-2M word
embeddings from fastText (Mikolov et al., 2018)
to search similar words. The dimensions of word
embedding $k$ and contextual representation $d$ are
set as 300 and 768. $w$ is set as 2. We follow
BERT$_{\text{BASE}}$ (Devlin et al., 2018) to set the num-
bers of layers (i.e., Transformer blocks) and self-
attention heads to 12.

### 4.2 Experimental Results

**Performance on identifying perpetuated to-
kens.** Table 3 shows the performance of DISP and
SC in discriminating perturbations. Compared to
SC, DISP has an absolute improvement by 35%
and 46% on SST-2 and IMDb in terms of F1-
score, respectively. It also proves that the con-
text information is essential when discriminating
the perturbations. An interesting observation is
that SC has high recall but low precision scores
for character-level attacks because it is eager to
correct misspelling while most of its corrections
are not perturbations. Conversely, DISP has more
balances of recall and precision scores since it
is optimized to discriminate the perturbed tokens.

For the word-level attacks, SC shows similar low
performance on both random and embed
attacks while DISP behaves much better. Moreover, DISP
works better on the random attack because the em-
beddings of the original tokens tend to have no-
ticeably greater Euclidean distances to randomly-
picked tokens than the distances to other tokens.

**Defense Performance.** Table 4 reports the ac-
curacy scores of all methods with different types of
adversarial attacks on two datasets. Compared to
the baseline BERT model, all of the meth-
ods alleviate the performance drops. All meth-
ods perform better on blocking character-level at-
tacks than word-level attacks because word-level
attacks eliminate more information. For the base-

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Metric</th>
<th>Character-level Attacks</th>
<th>Word-level Attacks</th>
<th>Overall Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Insertion</td>
<td>Deletion</td>
<td>Swap</td>
</tr>
<tr>
<td>SST-2</td>
<td>SC</td>
<td>Precision</td>
<td>0.5087</td>
<td>0.4703</td>
<td>0.5044</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Recall</td>
<td>0.9369</td>
<td>0.8085</td>
<td>0.9151</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>0.6594</td>
<td>0.5947</td>
<td>0.6504</td>
</tr>
<tr>
<td></td>
<td>DISP</td>
<td>Precision</td>
<td>0.9725</td>
<td>0.9065</td>
<td>0.9552</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Recall</td>
<td>0.8865</td>
<td>0.8760</td>
<td>0.8680</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td><strong>0.9275</strong></td>
<td><strong>0.8910</strong></td>
<td><strong>0.9095</strong></td>
</tr>
<tr>
<td>IMDb</td>
<td>SC</td>
<td>Precision</td>
<td>0.0429</td>
<td>0.0369</td>
<td>0.0406</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Recall</td>
<td>0.9367</td>
<td>0.8052</td>
<td>0.8895</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td>0.0820</td>
<td>0.0706</td>
<td>0.0777</td>
</tr>
<tr>
<td></td>
<td>DISP</td>
<td>Precision</td>
<td>0.9150</td>
<td>0.8181</td>
<td>0.8860</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Recall</td>
<td>0.5068</td>
<td>0.4886</td>
<td>0.5000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F1</td>
<td><strong>0.6523</strong></td>
<td><strong>0.6118</strong></td>
<td><strong>0.6392</strong></td>
</tr>
</tbody>
</table>

Table 3: Performance of SC and DISP on identifying perpetuated tokens.
Table 4: The accuracy scores of methods with different adversarial attacks on two datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Attack-free</th>
<th>Character-level Attacks</th>
<th>Word-level Attacks</th>
<th>Overall Attacks</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Insertion</td>
<td>Deletion</td>
<td>Swap</td>
<td>Random</td>
</tr>
<tr>
<td>SST-2</td>
<td>BERT</td>
<td>0.9232</td>
<td>0.6498</td>
<td>0.6544</td>
<td>0.6774</td>
</tr>
<tr>
<td></td>
<td>SC</td>
<td>0.9174</td>
<td>0.9082</td>
<td>0.8186</td>
<td>0.8840</td>
</tr>
<tr>
<td></td>
<td>ADA</td>
<td>0.9174</td>
<td>0.8071</td>
<td>0.8071</td>
<td>0.8209</td>
</tr>
<tr>
<td></td>
<td>AT</td>
<td>0.9186</td>
<td>0.8186</td>
<td>0.8175</td>
<td>0.8025</td>
</tr>
<tr>
<td></td>
<td>DISP</td>
<td>0.9232</td>
<td>0.8278</td>
<td>0.8278</td>
<td>0.8301</td>
</tr>
<tr>
<td></td>
<td>DISP+SC</td>
<td>0.9197</td>
<td>0.9128</td>
<td>0.8681</td>
<td>0.9060</td>
</tr>
<tr>
<td>IMDb</td>
<td>BERT</td>
<td>0.9431</td>
<td>0.8586</td>
<td>0.8599</td>
<td>0.8568</td>
</tr>
<tr>
<td></td>
<td>SC</td>
<td>0.9193</td>
<td>0.8834</td>
<td>0.8794</td>
<td>0.8825</td>
</tr>
<tr>
<td></td>
<td>ADA</td>
<td>0.9393</td>
<td>0.8766</td>
<td>0.8765</td>
<td>0.8754</td>
</tr>
<tr>
<td></td>
<td>AT</td>
<td>0.8998</td>
<td>0.8958</td>
<td>0.8822</td>
<td>0.8787</td>
</tr>
<tr>
<td></td>
<td>DISP</td>
<td>0.9378</td>
<td>0.9310</td>
<td>0.9297</td>
<td>0.9301</td>
</tr>
<tr>
<td></td>
<td>DISP+SC</td>
<td>0.9395</td>
<td>0.9316</td>
<td>0.8772</td>
<td>0.9313</td>
</tr>
</tbody>
</table>

Table 5: The accuracy of DISP over different types of attacks on the SST-2 dataset with the tokens recovered by the perturbation discriminator and the embedding estimator trained on the IMDb dataset for robust transfer defense. Note that DISP \(^x\) indicates the framework is established on the dataset \(x\).

<table>
<thead>
<tr>
<th>Method</th>
<th>Insertion</th>
<th>Delete</th>
<th>Swap</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>0.6498</td>
<td>0.6544</td>
<td>0.6774</td>
</tr>
<tr>
<td>DISP(_{SST-2})</td>
<td>0.8278</td>
<td>0.8278</td>
<td>0.8301</td>
</tr>
<tr>
<td>DISP(_{IMDb})</td>
<td>0.8243</td>
<td>0.8197</td>
<td>0.8278</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Random</th>
<th>Embed</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>0.5385</td>
<td>0.6556</td>
<td>0.6351</td>
</tr>
<tr>
<td>DISP(_{SST-2})</td>
<td>0.7773</td>
<td>0.7784</td>
<td>0.8083</td>
</tr>
<tr>
<td>DISP(_{IMDb})</td>
<td>0.7623</td>
<td>0.7681</td>
<td>0.8005</td>
</tr>
</tbody>
</table>

Number of Attacks. Figure 4 shows the classification accuracy of all methods over different numbers of attacks, i.e., perturbations, for different types of adversarial attacks. Without using a defense method, the performance of BERT dramatically decreases when the number of attacks increases. With defense approaches, the performance drops are alleviated. Moreover, the relations between the performance of methods are consistent across different perturbation numbers. DISP+SC consistently performs the best for all of the cases when DISP outperforms all of the single methods for most of the situations. These results demonstrate the robustness of the proposed approach.

Robust Transfer Defense. In practice, we may not have access to the original training corpus of a prediction model. In the following, we investigate if the perturbation discriminator can transfer across different corpora. We first train the discriminator and the estimator on IMDb denoted as DISP\(_{IMDb}\) and then apply it to defend the prediction model on SST-2. Table 5 shows the experimental results of robust transfer defense. DISP\(_{IMDb}\) achieves similar performance as the performance of DISP\(_{SST-2}\) trained on the same training set. Hence, it shows that DISP can transfer the ability to recover perpetuated token across different sentiment corpora.

Case Study of Recovered Text. Table 6 lists four
Table 6: A case study of recovered tokens in SST-2. Note that Label and Pred represent the ground-truth label and the predicted label.

<table>
<thead>
<tr>
<th>#</th>
<th>Attacked Sentence</th>
<th>Recovered Token</th>
<th>Label</th>
<th>Pred</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Mr. Tsai is a very original artist in his medium, and what time is it there?</td>
<td>imaginative</td>
<td>positive</td>
<td>positive</td>
</tr>
<tr>
<td>2</td>
<td>Old-form moviemaking at its best.</td>
<td>best</td>
<td>positive</td>
<td>positive</td>
</tr>
<tr>
<td>3</td>
<td>My reaction in a word: disappointment.</td>
<td>that</td>
<td>negative</td>
<td>positive</td>
</tr>
<tr>
<td>4</td>
<td>a painfully funny ode to bad behavior.</td>
<td>painfully; silly; one</td>
<td>negative</td>
<td>positive</td>
</tr>
</tbody>
</table>

Table 7: The performance of DISP using ground-truth and recovered tokens over different types of attacks in SST-2. Result are in accuracy. Note that DISP_G denotes DISP using ground-truth tokens.

<table>
<thead>
<tr>
<th>Method</th>
<th>Insertion</th>
<th>Delete</th>
<th>Swap</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISP_G</td>
<td>0.8773</td>
<td>0.8681</td>
<td>0.8796</td>
</tr>
<tr>
<td>DISP</td>
<td>0.8278</td>
<td>0.8278</td>
<td>0.8301</td>
</tr>
<tr>
<td>Method</td>
<td>Random</td>
<td>Embed</td>
<td>Overall</td>
</tr>
<tr>
<td>DISP_G</td>
<td>0.7970</td>
<td>0.7924</td>
<td>0.8429</td>
</tr>
<tr>
<td>DISP</td>
<td>0.7773</td>
<td>0.7784</td>
<td>0.8083</td>
</tr>
</tbody>
</table>

Embedding Estimator. Although DISP is not required to recover the ground-truth perturbed tokens, the embedding estimator plays an important role to derive appropriate embedding vectors that obtain the original semantics. We first evaluate the performance of embedding estimator as a regression task. The RMSE scores of estimated embeddings are 0.0442 and 0.1030 in SST-2 and IMDb datasets, which are small enough to derive satisfactory tokens. To further demonstrate the robustness of the embedding estimator and estimated embeddings, we identify the perturbations with our discriminator and replace them with the ground-truth tokens. Table 7 shows the accuracy scores over different types of attacks in the SST-2 dataset. DISP and DISP_G denotes the recovery performance with our estimator and ground-truth tokens, respectively. More specifically, the accuracy of DISP_G presents the upperbound performance gained by the embedding estimator. The experimental results demonstrate the robustness of...
Linguistic Acceptability Classification. In addition to the task of sentiment analysis, we also evaluate the performance of DISP in linguistic acceptability classification. The Corpus of Linguistic Acceptability (CoLA) is a binary classification task. The goal of this task is to predict whether an English sentence is linguistically acceptable or not (Warstadt et al., 2018). Table 8 presents the accuracy scores of BERT and DISP on the CoLA dataset with one adversarial attack of each type. It is interesting that the original BERT is extremely vulnerable to the adversarial attacks. This is because the linguistic acceptability can be easily affected by perturbations. The experimental results also depict that DISP can significantly alleviate the performance drops. DISP is capable of blocking adversarial attacks across different NLP tasks.

<table>
<thead>
<tr>
<th>Method</th>
<th>Insertion</th>
<th>Delete</th>
<th>Swap</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>0.1160</td>
<td>0.1407</td>
<td>0.1806</td>
</tr>
<tr>
<td>DISP</td>
<td>0.5856</td>
<td>0.5684</td>
<td>0.6008</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Random</th>
<th>Embed</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT</td>
<td>0.0855</td>
<td></td>
<td>0.1209</td>
</tr>
<tr>
<td>DISP</td>
<td>0.4848</td>
<td></td>
<td>0.5502</td>
</tr>
</tbody>
</table>

Table 8: The accuracy scores of BERT and DISP over different types of attacks on the CoLA dataset for the task of linguistic acceptability classification. The accuracy score of BERT without any attack is 0.8519.

5 Conclusions

In this paper, we propose a novel approach to discriminate perturbations and recover the text semantics, thereby blocking adversarial attacks in NLP. DISP not only correctly identifies the perturbations but also significantly alleviates the performance drops caused by attacks.

Acknowledgment

We would like to thank the anonymous reviewers for their helpful comments. The work was supported by NSF DGE-1829071 and NSF IIS-1760523.

References


Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency.


