# TextShield: A Defense against Word-level Text Attack by Adaptive Word Importance

Lingfeng Shen*
China Telecom Research Institute
shenlingfengcau@outlook.com

JinPeng Chen†
China Telecom Research Institute
jpchen@bupt.edu.cn

Haiyun Jiang‡
Tencent AI Lab
haiyunjiang@tencent.com

Jie Hu§
China Telecom Research Institute
jiehu1@chinatelecom.cn

Chen Ying¶
China Agricultural University
chenying@cau.edu.cn

## ABSTRACT

In text mining, deep neural network (DNN) models have long been criticized for their vulnerability to adversarial attacks. In this paper, we consider the task of text classification and propose TextShield, a defense against word-level text attacks by leveraging deep learning explainability techniques. In our paper, we start from one principal question: how to distinguish adversarial and benign sentences? Then we firstly show a correlation between DNN explainability and adversarial attack, which links the two long-standing challenges of DNN: robustness and unexplainability. Then we propose a novel metric called adaptive word importance (AWI), which owns significantly better flexibility than existing word importance. Based on AWI, we propose TextShield, composed of several detectors and one corrector. The detectors aim at judging whether the input sentence is adversarial and the corrector rectifies the adversarial sentences to benign ones. The detectors' parameters are pre-trained by recognizing whether sentences are adversarial. Furthermore, the random forest integrates multiple detectors to form a stronger detector for the adversarial detection of an input sentence. Comprehensive experiments show that using TextShield as the defense of adversarial attacks, the neural classification models (e.g., TextCNN, LSTM, and BERT) consistently achieve 8.0%, 4.8%, and 2.8% higher robust accuracy than state-of-the-art defense methods.

## CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Neural networks**.

## KEYWORDS

Adversarial Defense, Natural Language Processing

## 1 INTRODUCTION

Deep Neural Networks (DNNs) have obtained great progress in text mining. Nevertheless, researchers have observed that DNNs are vulnerable to adversarial examples, in which the clean data are modified imperceptibly by humans but could mislead DNN-based models to output wrong predictions [9, 10, 21, 23]. In real-world text mining scenarios, adversary sentences often lead to security and safety concerns, so research on defense algorithms against such adversaries is urgently needed.

Text classification is a mainstream task for the robustness check of text attacks. Furthermore, the most common and effective attack for text classification is word-level attack [8, 19, 35, 40], which is usually implemented by adding, deleting, or substituting words within a sentence. Such an attack often brings catastrophic performance degradation to DNN-based text classification models. Therefore, we choose to study defending against word-level attacks[1] in this paper.

In general, current defense methods against word-level text attacks can be categorized into three paradigms: (1) model enhancement-based [6, 14, 34, 44], (2) certified robustness-based [13, 17, 39], and (3) detection-based [22] defense.

- Model enhancement-based defense [6, 14, 34, 44] mainly focuses on adversarial training by further retraining the target model with adversarial examples. To guarantee the effectiveness of defense, this paradigm relies on the diversity of additional training data.
- Certified robustness-based defense [13, 17, 39] calculates the boundary of word-level attack, and if a model fits in the boundary, it is robust no matter how adversarial examples are created. However, the calculation of certified robustness is largely affected by models and testing data.
- Detection-based defense [22] is a new architecture for defense which follows a detect-correct pipeline. The detector tries to judge whether the sentence is adversarial, and the corrector corrects the sentence if it is adversarial. In [22], the detector is built by exploiting the frequency difference between the original word and the adversarial word.

In this paper, we work on defense under the detection-correction pipeline and focus on the key issue of detection-based defense: developing effective detectors. Though word frequency [22] is shown to be effective in adversarial detection, it will be much less effective when the frequency difference between the original word and the adversarial word is not distinctive, which indicates frequency-based detector is extremely insufficient. Thus, the goal of this paper is to build a stronger detector.

To this end, we consider one fundamental question: can we capture the differences between benign and adversarial sentences? Once we can capture the differences, distinguishing the two kinds of sentences becomes easier. Centering on this question, we design several toy experiments, showing that directly learning to distinguish benign and adversarial sentences failed to achieve promising results since benign and adversarial sentences are quite similar (e.g., only one-word change), making it difficult for DNNs to learn. In order to tackle such a problem, we make basic derivations and find that:

[1] Through the rest of the paper, text attack specifically refers to word-level text attack.

(1) important words are crucial to the power of text attacks. (2) the process of a text attack is related to saliency information which is a basic definition in the explainability of DNN.

Based on such discoveries, we define a novel metric called adaptive word importance (AWI) based on the saliency computation used in the explainability of DNN [2, 32, 33]. Specifically, for an input sentence, AWI measures the impact of each word on determining the label of the sentence in a model. The reason why we refuse existing word importance metrics (e.g., TF-IDF) is that they are model-agnostic, task-agnostic (e.g., 'excellent' is vital in sentiment classification but is unimportant in category classification.), and sample-agnostic (e.g., the important word in one sentence may be trivial in other sentences.), such properties fail to comprehensively reflect the importance of one word under a specific setting.

Based on AWI, we build our defense, TextShield, which consists of an ensemble detector and one corrector. The ensemble detector combined of four saliency-based detectors and one frequency-based detector by a random forest mechanism to give a final judge of whether an input sentence is adversarial. Then, the corrector rectifies the candidate words with high adversarial probabilities in the sentence by their high-frequency synonyms. Furthermore, the four saliency-based detectors compute AWI values using four saliency computation methods, respectively, and these detectors can complement well with each other for detecting adversarial sentences using the ensemble mechanism.

The main contribution of this paper are summarized as follows:

- We design TextShield, a detection-based defense against word-level attacks. TextShield is centered on word importance to capture the differences between benign and adversarial sentences.
- We define a novel metric called adaptive word importance (AWI) to connect adversarial robustness and saliency information and then leverage such a connection to design four saliency-based detectors. In the text mining field AWI first demonstrates a cross-cutting link between adversarial robustness and the explainability of DNN.
- Our proposed TextShield demonstrates superior performance than SoTA defense methods under various neural models on three text classification benchmarks and shows better defense against the word-level text attacks.

# 2 RELATED WORK

## 2.1 Defense against word-level text attack

Model enhancement is a popular way to improve the model's robustness against text attacks, and one most common way is adversarial training methods which re-train a victim model with additional adversarial examples to obtain an enhanced model. Alzantot [1], and Ren [29] generated adversarial examples by an attack method and re-trained models which are more robust against the attack. Unfortunately, the robustness of re-trained models is often determined by the diversity of adversarial examples used in re-training. As a result, adversarial training is vulnerable to unknown attacks, which is unrealistic in real-world scenarios. The relatively successful category is synonym encoding [6, 36], a stronger method to defend word-level attacks by setting the same embedding for different synonyms.

Besides, a stream of recent popular defense methods [12, 13] concentrate on certified robustness, which ensures models are provably robust to all potential word perturbations within the constraints. For example, IBP [13] performs catastrophically on large pre-trained language models (e.g., BERT). However, because of the extreme time cost in the training stage, certified robustness is difficult to be applied to complex models [13].

Recently, some work [22] tries to design detection-based defense methods. Mozes [22] used the frequency difference between the original word and the adversarial word to design detectors. Despite their success, the word-frequency-based defense is ineffective when the word frequency difference is not distinctive.

## 2.2 Word importance

The research of measuring word importance can be dated back to tf-idf [15, 20], and weights based on variants of word occurrence frequency [27, 30]. However, these methods have the following three limitations, which lead to less effectiveness in describing word importance: not model-specific, not task-specific, and not sample-specific. Thus, this paper proposes adaptive word importance that uses saliency computation methods to calculate word importance since there is a cross-cutting link between word saliency and adversarial robustness.

# 3 BACKGROUND

## 3.1 Adversarial text attack

In this section, we present some basic definitions used in our framework. In the text classification scenario, $X = \{w_1, w_2, \ldots, w_d\}$ is a sentence with $d$ words, and $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_d$ are the corresponding word embeddings. $X$ is labeled with a class $y \in \mathcal{Y}$ with $|\mathcal{Y}| = N$. Given a text classifier $F$, $F(X)$ denotes the probability distribution over the class set $\mathcal{Y}$ for the input sentence $X$. We select the class $y_j$ with the largest probability in $F(X)$ as the predicted label of $X$.

Generally, in a text attack setting, there is a *victim classifier* and a *text attack*.

- Victim Classifier: Given a set of training data in a text classification benchmark, a classifier $F(x)$ can be trained, where the sentence encoder is usually embodied as CNN [16], LSTM [11] or BERT [5].
- Benign Sentence: The original sentences in a text classification benchmark.
- Text Attack: For each benign sentence $X$ with a true label $y_j$, the text attack $G(\cdot)$ will generate an adversarial sentence $X^* = G(X)$ by substituting words $w_i$ in $X$ with $w_i^*$, satisfying: (1) $F(X^*)$ and $F(X)$ output different class labels with high probability and (2) $Dist\left(\mathbf{w}_i, \mathbf{w}_i^*\right) < \delta$. $Dist(\cdot, \cdot)$ is a distance metric and $\delta$ is a threshold to limit the distance between two words. For example, $Dist(\cdot, \cdot)$ can be the $l_2$ distance between two words' embedding. Intuitively, a small distance indicates that the two words are likely to be synonyms.
- Adversarial Sentences: The sentences generated by a text attack, which aims to mislead the victim classifier.

### 3.2 The Explainability of DNNs

In this section, we present some basic definitions used in the explainability of DNNs.

***Saliency***. A saliency map is a common way to explain deep learning, where saliency is a measure which quantifies the sensitivity of the model's output for input features. Specifically, a saliency value is a degree to which a DNN model considers an input feature to a class label, and a saliency map is a transparent colored heatmap overlaid on the original input. Such a map can be used to identify input features that are most salient in the sense that they cause a maximum influence on the model's output.

***Saliency Computation***. Back-propagation-based methods which have been widely used to compute saliency in the explainability of DNNs treat saliency as the gradient information of a DNN model. Due to the different ways to compute gradient information based on gradient signals passed from output to input during model training, there are various back-propagation-based saliency computation, such as vanilla gradient (VG), Guided Backpropagation (GBP) [32], Layerwise Relevance Propagation (LRP)[2, 3], Integrated Gradient (IG)[33]. E.g., in the vanilla gradient (VG) method, saliency is directly defined as the partial derivative of the network output concerning each input feature scaled by its value. Formally, in a standard classification scenario, given a classifier $F$, a sentence $X$ containing words $w_i$, the vanilla gradient (VG) method directly uses gradient information as the saliency of $w_i$ to the prediction $y_j$, defined as follows:

$$r_{ij}^{(VG)} = \frac{\partial F_{y_j}(X)}{\partial w_i} \tag{1}$$

## 4 INTUITION OF TEXTSHIELD

This section illustrates intuitions that motivate us to design TextShield. Firstly, we investigate whether it is simple to distinguish adversarial sentences from benign sentences. Then, we make a toy experiment to show that important words are crucial to attacks' effectiveness. Finally, we present basic derivations to show a cross-cutting connection between adversarial robustness and saliency information.

### 4.1 Are adversarial and benign sentences easy to be distinguished?

As illustrated in Sec 1, if we can capture the differences between adversarial sentences and benign sentences, then it is rather simple to detect adversarial sentences. In this part, we use three popular text attacks are applied to generate adversarial sentences: GSA [18], PWWS [29], GA [1]. Then, we choose a usual strategy which uses a sentence encoder (e.g., BERT) to produce the representation of a sentence, and makes a binary classification (predict a sentence as adversarial or benign) with the sentence representation, and apply [2]. However, such a strategy fails to distinguish adversarial and benign sentences, which is shown in Table 1.

The results illustrate that the performance of the usual strategy is nearly random guessing (50%). The adversarial sentences have tiny changes towards benign sentences, making them so close in the representation space. Therefore, the usual strategy and outlier detection methods [7] fail to effectively classify adversarial and benign sentences. Generally, adversarial and benign sentences are extremely difficult to distinguish, but is there any method that can help to distinguish them? Before answering the question, we present some empirical results below.

[2]We prepare three adversary detection settings under different attacks, and each setting owns 800 training samples and 200 test samples where every sample is labeled as 0 or 1, indicating whether it is adversarial. We use BERT-base as a sentence encoder and a MLP as a classifier to distinguish between adversarial and benign samples.

**Table 1: Test accuracy on adversarial sentence detection. Accuracy represents the model's accuracy on distinguish benign and adversarial sentences.**

| Accuracy | GA | PWWS | GSA | Avg |
|---|---|---|---|---|
| Normal | 58.5 | 56.3 | 60.1 | 58.3 |

### 4.2 Important words are crucial to attack's effectiveness

Intuitively, given a sentence, adding a perturbation to the most important word is more effective than adding to the unimportant one [29]. Therefore, focusing on that important words might be effective for adversarial detection, we design a toy experiment, which uses AGNews [43] as the dataset and applies three popular text attacks (GSA, PWWS, and GA) to generate adversarial sentences.

Specifically, we choose 300 adversarial sentences that successfully mislead the victim classifier. Thus, the original attack success rate is 100%. Then, for each of these 300 sentences, we calculate each word's saliency by the vanilla gradient method. Finally, we replace the words owning top 10% saliency $|r_{ij}|$ with [MASK] and feed the partially masked sentence to the victim classifier for evaluation. Saliency $|r_{ij}|$ is the one w.r.t the prediction $y_j \in [N]$ for input word $w_i$, and calculation of $|r_{ij}|$ is illustrated as follows:

$$|r_{ij}| = |\frac{\partial F_{y_j}(X)}{\partial w_i}|$$

where $F_{y_j}$ is the $j$-th element of confidence. A larger value of $|r_{ij}|$ indicates that the word $w_i$ is a more important word for classifying the sentence $X$ as the class $y_j$.

**Table 2: Accuracy after masking 10% most important words. 'Original' represents original accuracy while 'Modified' represents accuracy after masking.**

| Accuracy | GA | PWWS | GSA | Avg |
|---|---|---|---|---|
| Original | 0 | 0 | 0 | 0 |
| Modified | 31.7 | 30.3 | 29.8 | 30.6 |

As shown in Table 2, model's accuracy on such adversarial sentences is 0% while the modified accuracy increases to 30.6%. This toy experiment confirms our intuition that saliency $|r_{ij}|$ can be used to detect adversarial sentences.

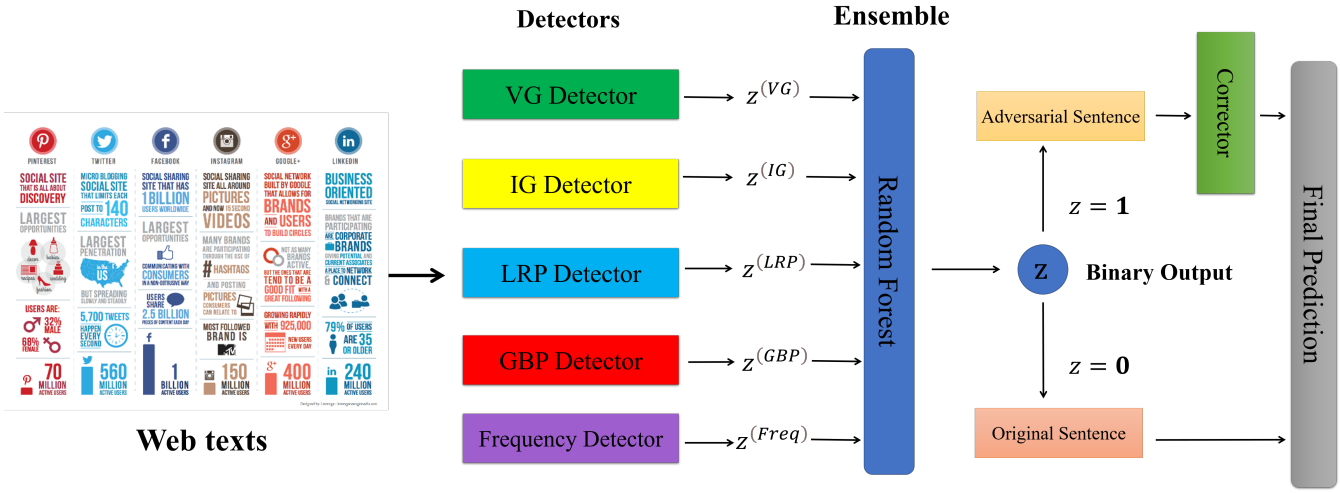



**Figure 1: The overview of TextShield.**

### 4.3 The connection between text attack and saliency information

The above empirical results demonstrate that the words with high saliency greatly influence text attack's power. In this part, we present a connection between text attacks and saliency information and show why saliency $|r_{ij}|$ is highly related to adversarial robustness.

For a benign sentence $X$ and a victim classifier $F$, the objective of a text attack which generates an adversarial sentence $X^*$ with a wrong prediction $y^{(j)}$ is defined as follows:

$$\arg\min \mathcal{L}\left(F_{y_j}\left(X^*\right), y^{(j)}\right) \tag{2}$$

where $\mathcal{L}$ is a loss function (e.g., cross-entropy loss). In a vanilla gradient-based attack [9], Eq 2 is optimized by a gradient descent method.

In each iteration of the attack, word $w_i$ is substituted with a word whose embedding is closest to $w_i^{'}$, and such an operation can probably fool the classifier. In other words, word embedding $\mathbf{w_i}$ is perturbed with a step rate $r_1$ by the attack, which is defined as follows:

$$w_i^{'} = w_i - r_1 \frac{\partial \mathcal{L}(F_{y_j}(X), y^{(j)})}{\partial w_i} \tag{3}$$

In ease of clarification, we regard the word embeddings as continuous variables. Then, we further derive the term $w_i - r_1 \frac{\partial \mathcal{L}(F_{y_j}(X), y^{(j)})}{\partial w_i}$, and rewrite it as follows:

$$w_i - r_1 \frac{\partial \mathcal{L}(F_{y_j}(X), y^{(j)})}{\partial F_{y_j}(X)} \cdot |r_{ij}| \tag{4}$$

As shown in Eq. (4), words with larger $|r_{ij}|$ are perturbed more severely by text attacks. Motivated by such a connection between text attacks and DNN's saliency information, we see that the term $|r_{ij}|$ ca'sn be helpful for adversarial detection. In one word, saliency $|r_{ij}|$ may serve as keys for adversarial sentence detection. Such derivations also match with our toy experiments in Sec 4.2.

### 4.4 Adaptive word importance

Since modifying on important words can decrease the power of text attacks (see Sec 4.2), we design a word importance metric to reflect the importance of a word. Considering that previous word importance metrics fail to adapt different sentences[3], different models, and different classification tasks[4], we design a novel metric called adaptive word importance (AWI) based on saliency information.

Definition 1. *Given sentence $X$ with prediction $y_j$ from text classifier $F$, the adaptive word importance (AWI) $r_{ij}$ of word $w_i$ in the sentence is defined as follows:*

$$R_{ij} = |\Delta(F, w_i, y_j)| \tag{5}$$

*where $\Delta$ is a saliency computation method, e.g., VG, IG [33].*

For sentence $X$, an AWI matrix $R$ is used to represent the importance of words in the sentence, where the $(i, j)$-th element $R_{ij}$ describes how important word $w_i$ is towards vim $F$ with prediction $y_j$. Thus, AWI $R_{ij}$ reflect the importance of a word in different settings (sentences, models, and classification tasks), and larger $R_{ij}$ indicates higher importance.

## 5 METHODOLOGY

### 5.1 Overview

Figure 1 shows the overview framework of our defense, TextShield, *which is like a shield in front of the victim classifier.* Generally, TextShield has two principal components: Ensemble Detector and Corrector. Given an input sentence $X$, $X$ is fed to the victim classifier $F$ and get prediction $y_j$. Then $X$ is filtered by TextShield which uses an ensemble detector to judge whether $X$ is benign or adversarial based on prediction $y_j$. If $X$ is benign, it will be fed to $F$ directly. Otherwise, A corrector is used to recover $X$ to $X^{(c)}$ and then feed $X^{(c)}$ to $F$ for prediction.

[3] the same word's meaning can be significantly different in different sentences

[4] the importance of a word is different in specific tasks. For example, 'happy' is important in sentiment classification (positive or negative) but remains unimportant in news classification (politics or sports).

***Ensemble detector***. As shown in Figure 1, the ensemble detector consists of two kinds of detectors: four **saliency-based detectors** and one **frequency-based detector**. In our framework, we take the random forest algorithm [4] to integrate the five detectors.

***Corrector***. If an input sentence $X$ is predicted as adversarial by the ensemble detector, the corrector will generate a new benign sentence $X'$ based on $X$ by substituting specific words with their frequent synonyms. Here, the specific words refer to the ones with high word importance. Then, the new sentence $X'$ will be fed into the classifier for prediction.

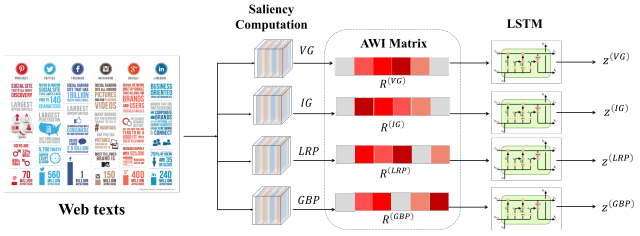



**Figure 2: The overview of the saliency-based detectors, and there are two stages in each detector. I: generate AWI matrix by specific saliency computation methods. II: feed the AWI matrix to LSTM and obtain the binary signal $z$.**

## 5.2 Saliency-based detectors

To utilize saliency information, we design detectors based on adaptive word importance (AWI) defined in Sec 4.4. As shown in Figure 2, given an input sentence, a detector firstly outputs an AWI matrix by a saliency computation method $\mathcal{I}$, and then produces a signal $z$ to indicate whether the sentence is adversarial or not. Specifically, four saliency computation methods are used to get the AWI matrix: vanilla gradient (VG), guided backpropagation (GBP) [32], layer-wise relevance propagation (LRP) [2], and integration gradient (IG) [33].

***Vanilla Gradient (VG)***. For text classifier $F$, the AWI of word $w_i$ in sentence $X$ to the prediction $y_j$ is calculated by the vanilla gradient as follows:

$$\mathcal{R}_{ij}^{(VG)} = \frac{\partial y_j}{\partial w_i} \tag{6}$$

where $\mathcal{R}^{(VG)} \in \mathbb{R}^{d \times N}$ is the AWI matrix, and the $(i, j)$-th element of $R$ is the AWI of word $w_i$ to the class $j$. In fact, when calculating the AWI, $y_j$ is a scalar and $w_i \in \mathbb{R}^{1\times k}$, where $k$ is dimension of word embedding. Thus, $\frac{\partial y_j}{\partial w_i}$ is a vector with the form $\mathbb{R}^{1\times k}$, and an average operation is made to obtain $\mathcal{R}_{ij}^{(VG)}$. For brevity, we omit the operation in our paper.

***Guided Backpropagation (GBP)***. Based on the idea of vanilla gradient, GBP [32] only chooses the 'positive' ($> 0$) gradient information instead of 'negative' ($< 0$) gradient information. Positive gradients indicate supporting the prediction, while negative ones mean the opposite. Such an operation enables us to capture what classifier $F$ learns but discards anything that $F$ does not learn. Specifically, when propagating the gradient, GBP [32] sets all the negative gradients to 0, and finally calculates the gradient of input towards prediction. We take the final outputs of GBP [32] as the importance matrix $\mathcal{R}^{GBP}$. Details of GBP can refer to [32].

***Layerwise Relevance Propagation (LRP)***. GBP owns obvious drawbacks called gradient saturation [2, 3] that the gradients computed by GBP are always zeros under some cases. Therefore, Layerwise Relevance Propagation (LRP) [2] is proposed. LRP points out this problem and highlights the importance of a reference input besides the target input to calculate the gradient. The LRP method calculates a weight for each connection in the gradient chain. It uses a reference input instead of roughly setting the weights as 0 or 1 like GBP. The reference input is a 'neutral' input and will be different in different tasks (e.g., blank images in CV field or zero embedding in NLP field). Specifically, it aims at decomposing classifier's confidence to calculate each variable's (e.g., word, pixel) contribution since it regards the confidence of a sum of the gradient of each input variable. Finally, we take such gradients as $\mathcal{R}_{ij}^{LRP}$. Details can refer to [32].

***Integrated Gradient (IG)***. Some works [33] pointed out that LRP is calculating the discrete gradient. However, the chain rule does not hold for discrete gradients thus LRP has essential drawbacks. The integrated gradient (IG) [33] calculates the influence of word $w_i$ to prediction $y_j$ by integration of vanilla gradient, defined as follows:

$$\mathcal{R}_{ij}^{(IG)} = \left(\mathbf{w_i} - \mathbf{w_i}'\right) \times \int_0^1 \frac{\partial F_{y_j}\left(X, \mathbf{w_i}' + \alpha\left(\mathbf{w_i} - \mathbf{w_i}'\right)\right)}{\partial \mathbf{w_i}} \, \mathrm{d}\alpha \tag{7}$$

where $w_i'$ is a zero embedding and $F_{y_j}(X, \mathbf{w_i}' + \alpha(\mathbf{w_i} - \mathbf{w_i}')$ means replace $w_i$'s embedding $\mathbf{w_i}$ with $\mathbf{w_i}' + \alpha(\mathbf{w_i} - \mathbf{w_i}')$.

***Training of saliency-based detector***. We train one LSTM to classify sentence $X$ as a benign or adversarial sentence for each detector. The LSTM's input is AWI matrices $\mathcal{R}_{\cdot j}$ generated by the different saliency computation methods (i.e., VG), which is defined as follows:

$$\begin{aligned} z^{(VG)} &= \mathrm{LSTM}_1\left(\mathcal{R}_{\cdot j}^{(VG)}\right), \quad & z^{(IG)} &= \mathrm{LSTM}_2\left(\mathcal{R}_{\cdot j}^{(IG)}\right) \\ z^{(GBP)} &= \mathrm{LSTM}_3\left(\mathcal{R}_{\cdot j}^{(GBP)}\right), \quad & z^{(LRP)} &= \mathrm{LSTM}_4\left(\mathcal{R}_{\cdot j}^{(LRP)}\right) \end{aligned} \tag{8}$$

where $z$ is confidence to judge whether the sentence is adversarial.

Since LSTMs are embeded in our TextShield, and their parameters are not trained. Specifically, we design a novel pre-train task called adversary detection to determine the parameters of LSTMs.

For a benign sentence set $\mathcal{S}_x = \{X_1, \ldots, X_N\}$, we use adversarial attacks to generate and select adversarial sentences that successfully fools the classifier. The training of TextShield is a binary classification task (adversarial sentences vs. benign sentences) is composed of two stages. The first stage is to train the LSTMs for saliency-based detectors, and each LSTM is trained independently. After training the LSTMs, we train the random forest in the second stage with the same setting. The detailed settings are reported in Sec 6.4.

## 5.3 Frequency-base detectors

Our frequency-based detector converts a sentence $X$ into $X^\circ$ by replacing random words with their frequent synonyms. We denote that $w_j$ is $w_i$'s synonym if their word embeddings satisfy $||\mathbf{w_i} -$

$\mathbf{w_j}||^2 \leq \delta$, where $\delta$ is a hyper-parameter, and we let $\mathcal{S}(w_i) = \{w : ||\mathbf{w_i} - \mathbf{w}||^2 \leq \delta\}$ denote the synonym set for $w_i$.

Firstly, we define a function $\phi(w_i, w_j) = \frac{p(w_i)}{p(w_j)}$ that indicates the frequency difference between two words $w_i, w_j$, where $p(w_i)$ means the frequency of $w_i$ in the training corpus. Then, frequency-based detector generates a sentence $X^\circ$ from $X$ by substituting random 10% words $w_i$ with more frequent words from $\mathcal{S}(w_i)$. Specifically, for each word $w \in \mathcal{S}(w_i)$ we select word $w$ with highest frequency: $w = \text{argmin}_{w \in \mathcal{S}(w_i)} \phi(w_i, w)$ and substitute $w_i$ with $w$. Therefore, $X^\circ$ is obtained after the substitution. Finally, the frequency-based detector outputs a value $z^{freq}$ to indicate whether the sentence is adversarial. For a label $y_j$ of $X$ and a threshold $\alpha \in [0, 1]$, the sentence $X$ is considered adversarial if $F(X)_{y_j} - F(X^\circ)_{y_j} > \alpha$. The sensitivity towards threshold $\alpha$ is recognized as sensitivity to word frequency [22], which will be discussed in the ablation study.

## 5.4 Ensemble mechanism

After designing four saliency-based detectors and one frequency-based detector, we use a random forest to integrate the five detectors as an ensemble detector, which is defined as follows:

$$z = RF\left(z^{(freq)}, z^{(VG)}, z^{(IG)}, z^{(GBP)}, z^{(LRP)}\right) \tag{9}$$

The output $z$ is the final judgment for adversarial detection; if $z < 0.5$, the sentence will be recognized as a benign one, and $z > 0.5$ means the opposite. With such an ensemble mechanism, the ensemble detector is more robust than a single detector when facing various kinds of text attacks.

## 5.5 Corrector

If a sentence is an adversarial one, the sentence will be fed to the corrector. As discussed above, given an importance matrix, larger $\mathcal{R}_{ij}$ means the word $w_i$ is more important for classifying the sentence as $y_j$. Specifically, $\mathcal{R}_{ij}$ is $\mathcal{R}_{ij}^{VG}$ in the corrector. If the ensemble detector recognizes the input sentence as an adversarial one, then the prediction class $j$ is possibly inaccurate, so the words with large $\mathcal{R}_{ij}$ are possible to be perturbed by attacks to fool the classifier. Therefore, the corrector corrects the words with a large $\mathcal{R}_{ij}$ in the adversarial sentence. Specifically, for an adversarial sentence $X^*$ with its prediction $y_j$, the corrector regards a word as a suspect when its $\mathcal{R}_{ij}$ exceeds a threshold, which is defined as follows:

$$\mathcal{R}_{ij} > \beta \cdot (\mathcal{R}_{\max} - \mathcal{R}_{\min}) + \mathcal{R}_{\min} \tag{10}$$

where $\mathcal{R}_{\max}$ and $\mathcal{R}_{\min}$ are the largest and smallest values in $\mathcal{R}$, and $\beta$ is a hyper-parameter to control the ratio of suspect. For a suspect $w_i$, the corrector substitute $w_i$ with the most frequent word $w' \in \mathcal{S}(w_i)$. Then the sentence is fed to the victim classifier.

# 6 EXPERIMENTS

## 6.1 Datasets and text attacks

We select three popular benchmarks in text classification, which are widely used both in text attack and defense: **AG's News** [43], **Yahoo! Answers** [43], and **IMDB** [26]. For word-level text attacks, we select four adversarial attacks, which are widely used in the stream of adversarial robustness in the NLP field: GSA [18], PWWS [29], GA [1] and IGA [37].

## 6.2 Baselines

We choose four defense methods as the baselines: (1) adversarial training (AT) [9]: the most common defense. (2) IBP [13]: the most representative method in certified robustness paradigm. (3) synonym encoding method (SEM) [36]: the SoTA defense method against word-level attack. (4) frequency-based word substitution (FGWS) [22]: the SoTA method in the detection paradigm.

## 6.3 Victim classifier

To validate that TextShield is effective across different victim models, we use TextCNN [16], LSTM [11], and Bert [5] as victim classifiers. For TextCNN and LSTM, we use Glove [25] for the pre-trained embedding with a size of 300. Specifically, TextCNN has three convolutional layers with the filter sizes of 3, 4, and 5, a dropout layer (dropout rate = 0.2), and a fully connected layer. The LSTM-based victim model has two layers, each with 128 cells. Bert is 'Bert-base' with the checkpoint from Huggingface. Moreover, for a fair comparison, as the previous work [31] reported that BERT is too difficult to be tightly verified by IBP, we do not use IBP on BERT.

## 6.4 Training settings

To train TextShield, we need to use text attacks to prepare adversarial examples. We use adversarial attacks to generate and select 2000 adversarial sentences that successfully attacks the model, equally generated by GA, IGA, GSA and PWWS. The four attacks are implemented by [41], and their settings are the same as their original papers, respectively. As for hyper-parameter settings, $\beta$ is set to 0.4 and $\delta$ is set to 0.5. The used LSTM has two LSTM layers where each layer has 128 LSTM cells and a fully-connected layer. Random forest is implemented by [24], the max depth is set as 3 and other parameters are default, considering that $z^{(freq)}$ is a discrete feature, we use a one-hot encoder [24] to transform. As for AT, we select 2000 adversarial sentences per class that successfully attack the model for adversarial training, which is the same as TextShield. Other baselines' settings are kept the same with their original papers.

## 6.5 Defence of TextShield

We sample 1000 benign sentences from each benchmark during evaluation and use the above text attacks to generate 1000 adversarial sentences. Then we evaluate the performance of victim models with or without defense methods. The more powerful the defense method is, the better classification accuracy on adversarial sentences is. Table 3 illustrates the performances of various defense baselines on benign sentences and adversarial sentences.

When evaluated on the benign sentences, adversarial training (AT) can improve the classification accuracy of most models on three datasets, which indicates AT can also enhance the generalization of training. IBP achieves much lower accuracy on benign data due to its high complexity and relatively loose bounds. Our proposed method, TextShield, achieves an accuracy close to normal training, with a good trade-off between robustness and generalization. Such a trade-off is usual for attack/defense in the CV field, which has been theoretically investigated [42].

**Table 3: The accuracy (%) of various models on three benchmarks, on benign sentences and adversarial sentences generated by different attacks. The numbers in bold indicate the best performance with the statistical significance threshold of p-value 0.05 in the paired t-test. No: benign sentences, NT: Normal Training, AT: Adversarial Training, TS: TextShield**

| Dataset | Attack | TextCNN | | | | | | LSTM | | | | | | BERT | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | NT | AT | IBP | FGWS | SEM | TS | NT | AT | IBP | FGWS | SEM | TS | NT | AT | FGWS | SEM | TS |
| IMDB | No | 88.7 | **89.1** | 78.6 | 84.7 | 86.8 | 88.5 | 87.3 | **89.6** | 79.5 | 80.9 | 86.8 | 85.5 | 92.3 | **92.5** | 88.7 | 89.5 | 88.0 |
| | GSA | 13.3 | 16.9 | 72.5 | - | 66.4 | **75.5** | 8.3 | 21.1 | 70.0 | - | 72.2 | **75.0** | 24.5 | 34.4 | - | 89.3 | **92.0** |
| | PWWS | 4.4 | 5.3 | 72.5 | 68.5 | 71.1 | **76.5** | 2.2 | 3.6 | 70.0 | 57.1 | 77.0 | **81.5** | 40.7 | 52.2 | 48.9 | 89.3 | **92.5** |
| | GA | 7.1 | 10.7 | 71.5 | 70.2 | 71.8 | **77.5** | 2.6 | 9.0 | 69.0 | 54.9 | 77.0 | **82.0** | 40.7 | 57.4 | 55.0 | 89.3 | **92.0** |
| | IGA | 2.3 | - | 65.0 | - | 72.4 | **78.0** | 2.7 | - | 64.5 | - | 73.8 | **78.7** | 28.3 | - | - | 85.5 | **91.0** |
| AG's News | No | **92.3** | 92.2 | 89.4 | 90.2 | 89.7 | 90.5 | 92.6 | 92.8 | 86.3 | 87.5 | 90.9 | 91.0 | 94.6 | **94.7** | 93.5 | 94.1 | 93.6 |
| | GSA | 45.5 | 55.5 | 86.0 | - | 80.0 | **88.5** | 35.0 | 58.5 | 79.5 | - | 85.5 | **88.5** | 66.5 | 74.0 | - | 88.5 | **89.0** |
| | PWWS | 37.5 | 52.0 | 86.0 | 76.5 | 80.5 | **89.0** | 30.0 | 56.0 | 79.5 | 75.5 | 86.5 | **89.2** | 68.0 | 78.0 | 72.3 | 88.5 | 88.0 |
| | GA | 36.0 | 48.0 | 85.0 | 72.5 | 80.5 | **88.5** | 29.0 | 54.0 | 76.5 | 80.5 | 85.0 | **87.5** | 58.5 | 71.5 | 75.0 | 88.5 | **90.5** |
| | IGA | 30.0 | - | 86.0 | - | 80.0 | **89.0** | 26.5 | - | 79.5 | - | 85.5 | **89.0** | 45.5 | - | - | 88.5 | **90.0** |
| Yahoo! Answers | No | 68.4 | **69.3** | 64.2 | 65.0 | 65.8 | 66.0 | 71.6 | **71.7** | 51.2 | 68.0 | 69.0 | 68.5 | **77.7** | 76.5 | 75.8 | 76.2 | 75.9 |
| | GSA | 19.6 | 20.8 | 61.0 | - | 49.4 | **64.5** | 27.6 | 30.5 | 30.0 | - | 48.6 | **64.0** | 31.3 | 41.8 | - | 66.8 | **70.6** |
| | PWWS | 10.3 | 12.5 | 61.0 | 52.5 | 52.6 | **64.0** | 21.1 | 22.9 | 30.0 | 50.2 | 54.9 | **64.0** | 34.3 | 47.5 | 50.6 | 66.8 | **70.0** |
| | GA | 13.7 | 16.6 | 61.0 | 53.0 | 59.2 | **64.5** | 15.8 | 17.9 | 30.5 | 45.5 | 66.2 | **67.0** | 15.7 | 33.5 | 55.5 | 66.4 | **70.0** |
| | IGA | 3.5 | - | 59.0 | - | 61.0 | **65.5** | 5.5 | - | 31.5 | - | 62.5 | **66.5** | 7.0 | - | - | 62.0 | **66.5** |

Although NT and AT achieved high accuracy on benign sentences, the accuracy of both NT and AT significantly decrease under text attacks on adversarial sentences. For NT, the accuracy drop exceeds 70%, 50%, and 50% on the three datasets. At the same time, AT is extremely powerless to defend against these attacks, especially for PWWS and GA. Specifically, AT can only improve accuracy by $5-10\%$ on adversarial sentences, far below the requirements of modern NLP models for robustness. One possible reason is that AT requires data diversity [29], and limited sentences are unable to meet the requirements.

When evaluated on the adversarial sentences, as for IBP, it should be noted that the IBP-like methods [28, 38] which were first proposed in the CV domain, is more suitable for CNN and does not perform excellently on LSTM. Moreover, IBP is difficult to train and does not perform perfectly on benign or adversarial examples. FGWS, on the other hand, focuses only on word frequencies, which is not sufficient to detect adversarial samples well. SEM achieves good results by synonym encoding, and however, TextShield demonstrates stronger defense capacity than SEM in almost every setting. Specifically, TextShield achieves 8.0%, 4.8%, and 2.8% higher accuracy than SEM.

## 7 ABLATION STUDY

This section illustrates ablation studies of TextShield by answering four research questions:

- What is the impact of hyper-parameters of TextShield?
- What is the impact of detector individually?
- Will the saliency-based detector help adversary detection?
- How does the amount of training sentences influence the performance?

### 7.1 Analysis of hyper-parameters

We have three hyper-parameters in our method:(1) $\delta$: the threshold for $l_2$ distance between two words' embeddings. (2) $\beta$: the threshold for selecting suspects in the corrector. (3) $\alpha$: the threshold in the frequency-based detector.

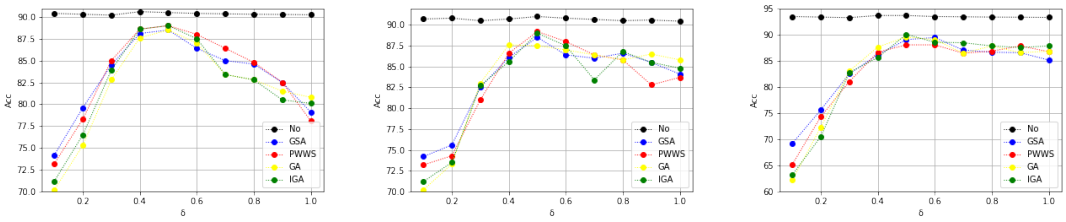
As shown in Figure 3, the accuracy on benign sentences (generalization) remains nearly unchanged, and we only need to select the hyper-parameters based on accuracy on adversarial sentences (robustness). Then, we investigate how the distance $\delta$ influences the performance of TextShield. When $\delta = 0$ or $\delta = 1$, the performance is not satisfactory because a small $\delta$ results in insufficient word synonyms, and a large $\delta$ may include words that are not semantically close. When $\delta < 0.5$, the performance increases continuously as $\delta$ increases. When $\delta$ exceeds 0.5, the accuracy starts to decrease rapidly. Therefore, we choose $\delta = 0.5$ to preserve the best robustness against text attacks.

Similarly, we study the impact of $\beta$ in the corrector. As shown in Figure 4, the performance achieves the best when $\beta$ is 0.4, and either a small or large $\beta$ will cause a significant performance drop, especially when $\beta$ is large. This is because $\beta$ serves to control the ratio of suspects in a sentence: when $\beta$ is 0, only the most important word is considered as a suspect, which may be inadequate; on the contrary, when $\beta$ is 1.0, every word will be viewed as a suspect, which is inappropriate.

Finally, the previous detection-based method (i.e., FGWS) is sensitive to the frequency difference (i.e., the threshold $\alpha$ ), so we compare the performance of TextShield and FGWS as $\alpha$ changes. As Figure 5 shows, TextShield achieves better and more stable performance than FGWS as $\alpha$ changes, especially when $\alpha$ is getting larger. Such results indicate that TextShield has less dependence on the word frequency due to the support of saliency-based detectors. Such results demonstrate that the saliency information significantly extends the defense ability of the detect-correct paradigm in adversarial text defense.

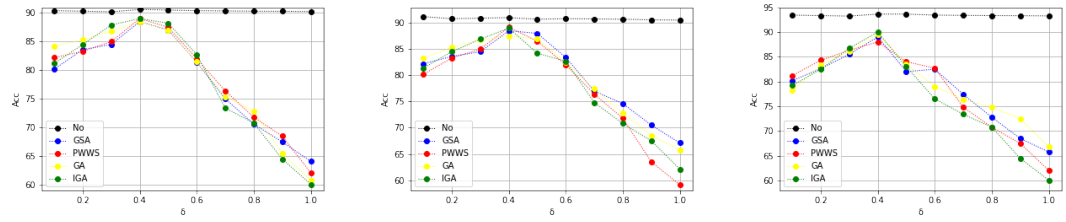
### 7.2 Analysis of ensembling

Since a crucial design of TextShield is the combination of various detectors to judge whether the input sentence is an adversarial one, we investigate the impact of such an ensembling strategy. Specifically, we verify the performance after removing some detectors, and the evaluation is launched on TextCNN on AGNews.




(a) TextCNN (b) LSTM (c) Bert

**Figure 3: Accuracy (%) of TextShield as $\delta$ varies from 0.1 to 1.0 for the three victim models on AG's News, $\delta$ is set as 0.5.**




(a) TextCNN (b) LSTM (c) BERT

**Figure 4: Accuracy (%) of TextShield as $\beta$ varies from 0.1 to 1.0 for the three victim models on AG's News, $\beta$ is set as 0.4.**

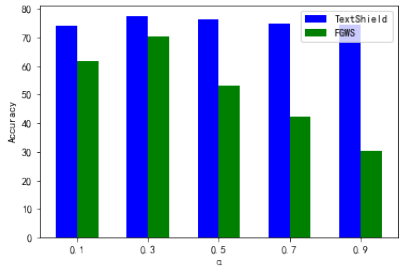



**Figure 5: Accuracy (%) of TextShield and FGWS on adversarial sentences as $\alpha$ changes.**

As shown in Table 4, removing any detector will cause a performance drop, demonstrating the effectiveness of combining various detectors with an ensemble mechanism. Specifically, based on the results, the importance of detectors can be ranked as follows: VG>Freq>LRP>IG>GBP. The results confirm our arguments that different detectors can complement well with each other. Furthermore, removing all the detectors shows a catastrophic performance drop, which shows the importance of the integration of the five detectors in the ensemble detector.

**Table 4: Accuracy (%) drop after removing some detectors. E.g., '-VG' means removing the VG detector in TextShield.**

| Removed detector | GSA | PWWS | GA | IGA |
|---|---|---|---|---|
| -VG detector | -8.9 | -7.9 | -6.4 | -7.0 |
| -IG detector | -6.8 | -4.5 | -3.0 | -4.5 |
| -LRP detector | -6.6 | -5.0 | -2.8 | -4.0 |
| -GBP detector | -7.4 | -2.5 | -4.1 | -5.1 |
| -Freq detector | -2.0 | -2.1 | -2.3 | -2.7 |
| -All detectors | -34.8 | -29.6 | -40.3 | -36.9 |

**Please note that** neither saliency computation methods nor combination methods of various detectors are our paper's focus. There are more sophisticated methods for saliency computation and detector combination, and such methods can also be applied for TextShield. In our paper, our main claim is that, in text mining, we can enhance the model's robustness by leveraging a cross-cutting relation between adversarial robustness and saliency information used in DNN's explainability, which is contrary to the conventional idea that text attacks should be defended by using discrete information (e.g., synonym encoding) instead of continuous information (e.g., saliency).

**Table 5: Test accuracy on adversarial sentence detection.**

| Accuracy | GA | PWWS | GSA | Avg |
|---|---|---|---|---|
| Normal | 58.5 | 56.3 | 60.1 | 58.3 |
| AWI | 80.4 | 82.0 | 81.1 | 83.2 |

## 7.3 Does AWI help adversary detection?

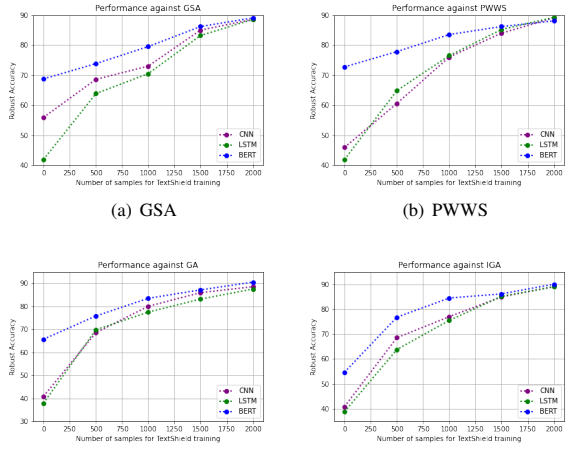
To examine the effectivity of AWI for distinguishing adversarial and benign sentences, we discard the detect-correct pipeline, and

use only the binary classifier in Sec 4.1 with the input: (1) Normal: sentence representation (encoded by BERT). (2) AWI: the AWI matrix (computed by VG). The experimental setting is the same as the one in Table 1, and the results are listed in Table 5, demonstrating that our AWI significantly helps with adversary detection.

## 7.4 Analysis of the number of sentences used in TextShield training

During the training process, we use the same number $K$ of adversarial sentences per class for TextShield. The results in the main paper are under the condition where $K = 2000$. This section illustrates the relation between TextShield and $K$. Specifically, we keep the same training settings mentioned in the main paper except for $K$. The performance of TextShield under different $K$ is shown in Figure 6.

As Figure 6 shows, the defense capacities of TextShield are enhanced as $K$ continuously increases since more samples in the training help the detectors recognize adversarial sentences better. Specifically, when $K$ is set as 500, TextShield still significantly outperforms the performance of AT when using 2000 adversarial sentences for training, further validating the effectiveness of TextShield.




(a) GSA (b) PWWS

(c) GA (d) IGA

**Figure 6: Accuracy (%) of TextShield on adversarial sentences generated by four attacks as $K$ changes.**

## 8 CONCLUSION

This paper proposes TextShield, a detect-correct pipelined defense against word-level text attacks. We design TextShield by exploring one fundamental question step by step: how to distinguish adversarial and benign sentences. Based on empirical findings and basic derivations, we propose AWI, adaptive word importance that reflects a word's importance in specific scenarios. Based on AWI, we design saliency-based detectors for TextShield, which significantly enhances the detect-correct paradigm defense. Comprehensive experiments demonstrate that TextShield is superior to SoTA baselines when defending against text attacks. TextShield achieves promising defense performance by leveraging a cross-cutting connection between adversarial robustness and explainability of DNN, and we believe it sheds light on the adversarial robustness field in text mining. Moreover, such a pipeline is not specific to text classification, and we will explore the capacity of TextShield on other tasks (e.g., QA, NLI) in the future.

## REFERENCES

[1] Moustafa Alzantot and et al. 2018. Generating Natural Language Adversarial Examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2890–2896.
[2] Sebastian Bach and et al. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.
[3] Alexander Binder and et al. 2016. Layer-wise relevance propagation for neural networks with local renormalization layers. In *International Conference on Artificial Neural Networks*. Springer, 63–71.
[4] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
[5] Jacob Devlin and et al. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*. 4171–4186.
[6] Dong and et al. 2021. Towards robustness against natural language word substitutions. *ICLR* (2021).
[7] Ruize Gao, , and et al. 2020. Maximum mean discrepancy is aware of adversarial attacks. *arXiv preprint arXiv:2010.11415* (2020).
[8] Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based Adversarial Examples for Text Classification. In *EMNLP*. 6174–6181.
[9] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* (2014).
[10] Chuan Guo and et al. 2019. Simple Black-box Adversarial Attacks. *Proceedings of Machine Learning Research* (2019).
[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[12] Po-Sen Huang and et al. 2019. Achieving Verified Robustness to Symbol Substitutions via Interval Bound Propagation. In *EMNLP*. 4083–4093.
[13] Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified Robustness to Adversarial Word Substitutions. In *EMNLP*. 4129–4142.
[14] Erik Jones, Robin Jia, Aditi Raghunathan, and Percy Liang. 2020. Robust Encodings: A Framework for Combating Adversarial Typos. In *ACL*. 2752–2765.
[15] Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation* (1972).
[16] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP*. 1746–1751.
[17] Ching-Yun Ko and et al. 2019. POPQORN: Quantifying robustness of recurrent neural networks. In *ICML*. PMLR, 3468–3477.
[18] Volodymyr Kuleshov and et al. 2018. Adversarial examples for natural language classification problems. (2018).
[19] Dianqi Li and et al. 2021. Contextualized Perturbation for Textual Adversarial Attack. In *NAACL*. 5053–5069.
[20] Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM Journal of research and development* 2, 2 (1958), 159–165.
[21] Aleksander Madry and et al. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *International Conference on Learning Representations*.
[22] Maximilian Mozes, Pontus Stenetorp, Bennett Kleinberg, and Lewis Griffin. 2021. Frequency-Guided Word Substitutions for Detecting Textual Adversarial Examples. In *EACL*. 171–186.
[23] Nicolas Papernot and et al. 2016. The limitations of deep learning in adversarial settings. In *2016 EuroS&P*. IEEE, 372–387.
[24] Fabian Pedregosa and et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.
[25] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. 1532–1543.
[26] Christopher Potts. 2010. On the negativity of negation. In *Semantics and Linguistic Theory*, Vol. 20. 636–659.
[27] Xiaojun Quan and et al. 2010. Term weighting schemes for question categorization. *TPAMI* 33, 5 (2010), 1009–1021.
[28] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Certified Defenses against Adversarial Examples. In *ICLR*.
[29] Shuhuai Ren and et al. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*. 1085–1097.
[30] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Vol. 39. Cambridge University Press Cambridge.
[31] Zhouxing Shi and et al. 2019. Robustness Verification for Transformers. In *International Conference on Learning Representations*.
[32] J Springenberg, Alexey Dosovitskiy, Thomas Brox, and M Riedmiller. 2015. Striving for Simplicity: The All Convolutional Net. In *ICLR (workshop track)*.
[33] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*. PMLR,

3319–3328.
[34] Boxin Wang and et al. 2020. InfoBERT: Improving Robustness of Language Models from An Information Theoretic Perspective. In *International Conference on Learning Representations*.
[35] Wenqi Wang and et al. 2019. Towards a robust deep neural network in texts: A survey. *arXiv preprint arXiv:1902.07285* (2019).
[36] Xiaosen Wang and et al. 2021. Natural Language Adversarial Defense through Synonym Encoding. In *Conference on Uncertainty in Artificial Intelligence*.
[37] Xiaosen Wang, Hao Jin, and Kun He. 2019. Natural Language Adversarial Attack and Defense in Word Level. (2019).
[38] Eric Wong and Zico Kolter. 2018. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*. PMLR, 5286–5295.
[39] Kaidi Xu and et al. 2020. Automatic perturbation analysis for scalable certified robustness and beyond. *NeurIPs* 33 (2020).
[40] Yuan Zang and et al. 2020. Word-level Textual Adversarial Attacking as Combinatorial Optimization. In *ACL*. 6066–6080.
[41] Guoyang Zeng and et al. 2021. Openattack: An open-source textual adversarial attack toolkit. In *ACL*. 363–371.
[42] Hongyang Zhang and et al. 2019. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*. PMLR, 7472–7482.
[43] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *NeurIPs* 28 (2015), 649–657.
[44] Yi Zhou, , and et al. 2021. Defense against adversarial attacks in nlp via dirichlet neighborhood ensemble. *ACL* (2021).