Frequency-guided word substitutions for detecting textual adversarial examples

Mozes, Maximilian; Stenetorp, Pontus; Kleinberg, Bennett; Griffin, Lewis D.

Publication date:
2020

Document Version
Early version, also known as pre-print

Frequency-Guided Word Substitutions for Detecting Textual Adversarial Examples

Maximilian Mozes **Pontus Stenetorp** **Bennett Kleinberg** **Lewis D. Griffin**
University College London
{m.mozes, p.stenetorp, l.griffin}@cs.ucl.ac.uk, bennett.kleinberg@ucl.ac.uk

Abstract

While recent efforts have shown that neural text processing models are vulnerable to adversarial examples, comparatively little attention has been paid to explicitly characterize their effectiveness. To overcome this, we present analytical insights into the word frequency characteristics of word-level adversarial examples for neural text classification models. We show that adversarial attacks against CNN-, LSTM- and Transformer-based classification models perform token substitutions that are identifiable through word frequency differences between replaced words and their substitutions. Based on these findings, we propose *frequency-guided word substitutions* (FGWS) as a simple algorithm for the automatic detection of adversarially perturbed textual sequences. FGWS exploits the word frequency properties of adversarial word substitutions, and we assess its suitability for the automatic detection of adversarial examples generated from the SST-2 and IMDb sentiment datasets. Our method provides promising results by accurately detecting adversarial examples, with F_1 detection scores of up to 93.7% on adversarial examples against BERT-based classification models. We compare our approach against baseline detection approaches as well as a recently proposed perturbation discrimination framework, and show that we outperform existing approaches by up to 15.1% F_1 in our experiments.

1 Introduction

Recent advancements in machine learning research uncovered the vulnerability of artificial neural networks to adversarial examples – carefully crafted perturbations of input data that lead a supervised learning model into making false predictions. While this phenomenon has initially been discovered in the visual domain (Szegedy et al., 2014; Goodfellow et al., 2015; Kurakin et al., 2017), it has been shown that natural language processing models are oversensitive to adversarial input perturbations for a variety of tasks as well (Papernot et al., 2016; Jia and Liang, 2017; Belinkov and Bisk, 2018; Glockner et al., 2018). Although various works propose different classes of attacks against neural text processing models demonstrating their oversensitivity to adversarial inputs (e.g. Alzantot et al. (2018); Ebrahimi et al. (2018); Ren et al. (2019); Jin et al. (2019); Yang et al. (2020)), little attention has thus far been paid to a more detailed understanding of what causes textual adversarial examples to be successful, and whether individual perturbations provide potential cues that help us to automatically identify them.

To better understand model oversensitivity in the context of text classification tasks, this work examines the word frequency properties of word-level textual adversarial perturbations, and provides empirical evidence that CNN-, LSTM, and BERT-based text classification models are oversensitive to low-frequency word substitutions triggered by adversarial attacks. Experimenting with four recently proposed attacks (Alzantot et al., 2018; Ren et al., 2019), we demonstrate that such attacks tend to implicitly replace individual words with less frequent ones. We then show that their effectiveness can be mitigated through simple frequency-guided manipulations of adversarial sequences. Specifically, we introduce *frequency-guided word substitutions* (FGWS), a detection method that identifies adversarial sequences by directly manipulating individual words in a text based on their word frequency properties. Our findings show that FGWS can effectively be used to detect adversarial perturbations, achieving F_1 scores of up to 93.7% on discriminating unperturbed and perturbed sequences against BERT-based classification models (Devlin et al., 2019) on the IMDb movie reviews dataset.

We compare the performance of FGWS to

DISP (Zhou et al., 2019), a recently introduced perturbation discrimination model that exploits contextualized word representations, demonstrating that incorporating contextual information is effective for this task. FGWS instead explicitly considers low-frequency words as potentially adversarial, and aims to mitigate their effectiveness through simple frequency-guided word substitutions. We show improved adversarial sequence detection performances of FGWS as compared to DISP, indicating that our approach accurately discriminates perturbations without relying on any contextual information. Specifically, we demonstrate that, despite representing a far simpler approach, FGWS improves upon DISP by up to 15.1% F_1 on differentiating between unperturbed and perturbed sequences.

2 Related work

There exists a wide variety of adversarial attacks demonstrating the oversensitivity of text classification models, achieved through sequence manipulations on a character-, word- and sentence-level (Gao et al., 2018; Eger et al., 2019; Tsai et al., 2019; Behjati et al., 2019), or a combination of those (Li et al., 2018; Liang et al., 2018; Lei et al., 2019).

Ebrahimi et al. (2018) demonstrate the vulnerability of neural text classification models by proposing an adversarial attack that manipulates individual characters based on information sourced from the model's gradients with respect to one-hot character input representations. However, such character-level manipulations potentially degrade the text, thereby providing cues that can be detected by word recognition models to mitigate the attacks' effectiveness (Pruthi et al., 2019).

Word-level attacks manipulate textual sequences by inserting, replacing or removing individual words to generate adversarial examples. Papernot et al. (2016) propose an attack against text classification models that replaces individual words in an input sequence by utilizing the model's gradients to identify the most effective adversarial word substitutions in the model's vocabulary. Although highly effective, one of the attack's disadvantages is that the perturbed sequence might lose its semantic and/or syntactic correctness. Recent works overcome this by generating adversarial examples that preserve the semantics and syntactic correctness of the sequence, using synonym sets and pre-trained language models to identify word substitutions that do not alter the sequence's semantics and fit in a word's context (Alzantot et al., 2018; Zhang et al., 2019; Ren et al., 2019). While most attacks are evaluated against CNN and LSTM classification models, Jin et al. (2019) have recently demonstrated that adversarial attacks can be effective against models based on pre-trained, contextualized word representations. Their approach generates adversarial examples against BERT-based classification models, thereby aiming to preserve both textual semantics and fluency.

Paraphrasing entire input sequences has also shown to serve as an effective tool for adversarial example generation. Iyyer et al. (2018) demonstrate this by proposing an encoder-decoder sequence paraphrasing model to generate adversarial paraphrases against models trained on sentiment and textual entailment datasets. Moreover, Ribeiro et al. (2018) present a method for auto-generating sets of semantics-preserving paraphrasing rules to generate adversarial examples, and demonstrate its effectiveness in sentiment analysis and visual question answering settings.

Existing efforts to overcome the effectiveness of textual adversarial examples and increase model robustness include adversarial training and data augmentation (Li et al., 2017; Jia and Liang, 2017; Ebrahimi et al., 2018; Ribeiro et al., 2018; Wang and Bansal, 2018; Ren et al., 2019; Jin et al., 2019; Cheng et al., 2019) as well as methods to achieve certified model robustness (Huang et al., 2019; Jia et al., 2019). Recently, Zhou et al. (2019) proposed an approach to detect adversarial sequences that exploits contextualized representations by utilizing BERT-based discrimination models to identify adversarial sequence tokens and restore the words that were replaced by an attack. In the present work, in contrast, we turn away from employing any contextual information and instead solely utilize word frequency characteristics to detect adversarially inserted words.

3 Generating textual adversarial examples

3.1 Setup

We denote a classification model by a function $f(X) \in \mathbb{R}^C$ that projects an input sequence X to a C-dimensional vector representing the unnormalized logits for each of the C possible classes. We represent a sequence as $X = x_1 x_2 \dots x_{n-1} x_n$,

where x_i denotes the i-th word in the sequence. We furthermore introduce the notation $f^*(X) \in \{1, \ldots, C\}$ representing the class label predicted by f with input X. In our adversarial setting, the adversary's goal is to identify an input sequence X' based on X such that $f^*(X') \neq f^*(X)$.

3.2 Adversarial attacks

We focus our experimentation on four recently proposed textual adversarial attacks, two of which are considered baselines. The first is based on genetic search (Alzantot et al., 2018) and the second utilizes word saliencies (Ren et al., 2019) to generate adversarial examples. Both methods have shown to be highly effective at attacking text classification models, and the former has been investigated in related work focusing on achieving certified robustness to word-level adversarial attacks (Jia et al., 2019). We additionally experiment with two baseline methods as introduced by Ren et al. (2019).

RANDOM. Our first baseline attack is a simple word substitution model that randomly selects words in an input sequence and replaces them with synonyms that are also randomly sampled from the set of synonyms related to the specific word. We adhere to Ren et al. (2019)'s realization by utilizing WORDNET (Fellbaum, 1998) to identify potential synonym substitutions for each selected word.

PRIORITIZED. Our second baseline samples words from a given input sequence and selects a substitution from each word's synonym set by finding the synonym that maximizes the change in prediction confidence on the true label of our input sequence. A word's synonym set is computed analogously to the RANDOM attack.

PWWS. We furthermore use the recently proposed *probability weighted word saliency* (PWWS) algorithm (Ren et al., 2019), a word-level adversarial attack based on synonym substitutions. For each word in the input sequence, the algorithm selects a set of synonym replacements from WORDNET and chooses the synonym yielding the highest difference in prediction confidence on the true class label after replacement. The algorithm furthermore computes the word saliency (Li et al., 2016a,b) for each input word and defines an importance ranking of word replacements based on these two indicators. The input sequence is then manipulated by perturbing words according to this order. PWWS pays special attention to named entities by ensuring that named entities selected for replacement in the input sequence are replaced with other named entities of the same type.

GENETIC. Lastly, we analyze an attack suggested by Alzantot et al. (2018), consisting of a population-based black-box mechanism that iteratively adds individual word-level perturbations to an input sequence to lead a model into misclassification. To achieve this, Alzantot et al. (2018) leverage a population-based genetic search algorithm that crafts a population of candidate perturbations in different generations. Each generation inherits the highest-performing perturbations from the previous generation and further manipulates an input sequence. The algorithm terminates when a successful perturbation has been found or the maximum amount of generations has been reached.

3.3 Classification models

We apply the proposed attacks to three classification models. The first is a word-based convolutional neural network (CNN) for sequence classification (Kim, 2014) that has been employed in existing works studying textual adversarial attacks (Lei et al., 2019; Jia et al., 2019; Tsai et al., 2019). For the second classification model, we follow Alzantot et al. (2018) and Ren et al. (2019) and employ a single layer *Long Short-Term Memory* (LSTM) network (Hochreiter and Schmidhuber, 1997). Both the LSTM and CNN are initialized with pre-trained GLOVE (Pennington et al., 2014) word embeddings. The third is a pre-trained BERT_{base} (Devlin et al., 2019) model fine-tuned for binary classification.

3.4 Datasets and performance details

We train all three classification models on two binary text classification datasets: the *Internet Movie Database* (IMDb) reviews dataset (Maas et al., 2011) and the *Stanford Sentiment Treebank* (SST-2) as introduced by Socher et al. (2013). Both datasets have been used in previous works related to textual adversarial example generation (Papernot et al., 2016; Alzantot et al., 2018; Zhang et al., 2019; Jia et al., 2019; Tsai et al., 2019; Ren et al., 2019; Huang et al., 2019; Zhou et al., 2019).

IMDb. The IMDb movie reviews dataset consists of 50,000 positive and negative movie reviews sourced from the IMDb website with a pre-defined split of 25,000 training and 25,000 test samples, where each sample is labeled as either positive or

negative. We hold out 1,000 samples from the training set for validation.

SST-2. The SST-2 dataset contains movie reviews annotated with binary sentiment labels. The dataset comes with a pre-defined split of 67,349 samples for training, 872 for validation and 1,821 for testing.

Dataset	Classifier	Acc.	Attack success rate			
			RANDOM	PRIORITIZED	GENETIC	PWWS
IMDb	CNN	87.2	6.8	84.2	81.0	89.9
	LSTM	87.4	6.5	91.4	84.1	95.4
	BERT_{base}	91.3	5.4	71.5	70.0	61.4
SST-2	CNN	84.2	6.2	49.0	78.3	65.3
	LSTM	83.8	5.9	45.5	74.2	61.5
	BERT_{base}	92.2	4.3	34.0	63.0	42.7

Table 1: Overview of the attack success rates (%) of all four attacks when applied to the CNN, LSTM and BERT_{base} classification models with respect to both datasets.

Model performances. On the IMDb movie reviews dataset, the CNN achieves an accuracy of 86.6%, the LSTM achieves 86.6% and BERT_{base} achieves 90.8%, all when evaluated on the 25,000 test samples. These performances are comparable to existing works (Gao et al., 2018; Zhang et al., 2019; Ren et al., 2019; Jin et al., 2019). On the SST-2 dataset, the CNN achieves 84.3%, the LSTM 83.9% and BERT_{base} 92.2% accuracy when evaluated on the 1,821 elements of the test set, which are also comparable to existing works (Socher et al., 2013; Devlin et al., 2019; Huang et al., 2019). A detailed description of model architectures, hyperparameters and training details can be found in Appendix A.

3.5 Attack performances

We utilize all four attacks on a randomly sampled subset of 2,000 sequences from the IMDb test set as well as the entire test set of SST-2. For the GENETIC attack, we follow Alzantot et al. (2018) by limiting the allowed number of word replacements to 20% of the length of the input sequence and employ the same threshold for the two baseline attacks (RANDOM and PRIORITIZED) as well. A detailed description of the implementation and parameter details for all attacks can be found in Appendix B.

The attack success rates can be found in Table 1. Acc. denotes the percentage of those sequences that were correctly classified by the respective classifier (since only those can be considered for the attack). The attack success rates then represent the fraction of successfully created adversarial examples (i.e. the predicted class changed after perturbation) with respect to all correctly classified sequences. The results indicate that while the RANDOM baseline fails to successfully generate adversarial sequences, the three other attacks create successful perturbations for a majority of the tested combinations of dataset and classification model.

4 Statistical word frequency analysis of textual adversarial examples

The attack performances as shown in Section 3.5 demonstrate that all three classification models are vulnerable to textual adversarial examples. In an attempt to identify common statistical characteristics of the adversarial examples crafted with the different attacks, we analyze the word frequencies of individual replaced words and their respective substitutions.

4.1 Comparing occurrence frequencies of adversarial substitutions

We compute the $\log_e$ occurrence frequencies of *(i)* all words in the test set that are eligible for replacement by the respective attacks (see Appendix B), *(ii)* all words that have been replaced by the respective attacks and *(iii)* all of their corresponding substitutions. We denote the $\log_e$ occurrence frequency as $\phi(x)$ for a given word x, defined as $\phi(x) = \log_e(1 + \phi_{abs}(x))$, where $\phi_{abs}(x) \in \mathbb{N}_0$ denotes the absolute occurrence frequency of word x in the training corpus.

Table 2 shows the resulting $\log_e$ frequencies for the specified words. In the two right-most columns we differentiate between all adversarially inserted words and only those that occur in the model's training corpus and are hence not *out-of-vocabulary* (OOV) tokens, since the word substitution frequencies might primarily be decreased by OOV tokens. Across all datasets, classification models and attacks, the replaced words are not less or even slightly more frequent than the average amount of replaceable words, but the substitutions are consistently less frequent. Specifically, we observe that apart from the RANDOM attack, all attacks tend to select words for replacement whose frequency is slightly above the mean $\log_e$ frequency of replaceable words, but all four attacks select substitutions whose frequencies are lower than those of the replaced words. This observation holds even when we only consider word substi-

Dataset	Classifier	Attack	Replaceable words	Replaced words	Substitutions	Substitutions (non-OOV)
IMDb	CNN	RANDOM	6.5 (2.0)	6.6 (2.0)	4.1 (2.7)	4.9 (2.2)
		PRIORITIZED	6.5 (2.0)	6.7 (1.9)	4.0 (2.7)	4.7 (2.3)
		GENETIC	6.1 (2.1)	6.4 (2.0)	3.6 (2.3)	3.8 (2.2)
		PWWS	6.5 (2.0)	6.8 (2.2)	4.2 (2.8)	4.8 (2.4)
	LSTM	RANDOM	6.5 (2.0)	6.6 (2.0)	4.2 (2.7)	4.9 (2.2)
		PRIORITIZED	6.5 (2.0)	6.8 (1.9)	4.1 (2.5)	4.8 (2.1)
		GENETIC	6.1 (2.1)	6.3 (1.9)	3.6 (2.2)	3.8 (2.1)
		PWWS	6.5 (2.0)	6.7 (2.0)	4.4 (2.5)	4.9 (2.1)
	BERT_{base}	RANDOM	6.5 (2.0)	6.6 (2.0)	4.1 (2.7)	4.9 (2.2)
		PRIORITIZED	6.5 (2.0)	6.8 (1.9)	4.3 (2.6)	4.9 (2.2)
		GENETIC	6.1 (2.1)	6.5 (2.0)	3.6 (2.3)	3.9 (2.1)
		PWWS	6.5 (2.0)	6.9 (2.3)	4.6 (2.7)	5.2 (2.3)
SST-2	CNN	RANDOM	4.6 (1.9)	4.6 (2.0)	2.6 (2.3)	4.0 (1.6)
		PRIORITIZED	4.6 (1.9)	4.8 (1.8)	2.8 (2.2)	3.9 (1.6)
		GENETIC	4.2 (2.0)	4.3 (1.7)	2.1 (1.9)	3.2 (1.4)
		PWWS	4.6 (1.9)	4.8 (2.1)	3.0 (2.4)	4.1 (1.8)
	LSTM	RANDOM	4.6 (1.9)	4.6 (1.9)	2.6 (2.3)	4.0 (1.6)
		PRIORITIZED	4.6 (1.9)	4.7 (1.8)	2.9 (2.1)	3.8 (1.6)
		GENETIC	4.2 (2.0)	4.2 (1.7)	2.2 (1.9)	3.2 (1.4)
		PWWS	4.6 (1.9)	4.8 (2.1)	3.2 (2.4)	4.2 (1.8)
	BERT_{base}	RANDOM	4.6 (1.9)	4.6 (1.9)	2.6 (2.3)	4.0 (1.5)
		PRIORITIZED	4.6 (1.9)	4.7 (1.9)	2.6 (2.3)	4.0 (1.6)
		GENETIC	4.2 (2.0)	4.4 (1.9)	1.9 (2.2)	3.6 (1.6)
		PWWS	4.6 (1.9)	4.8 (2.1)	3.1 (2.5)	4.3 (1.8)

Table 2: Average $\log_e$ frequencies of replaced words and their corresponding substitutions by attack, classifier and dataset. The shown values are the mean (and standard deviation) $\log_e$ frequencies for each setting. Replaceable words denotes the $\log_e$ frequencies of all words occurring in the tested sequences that are allowed to be replaced by the respective attack.

tutions that occur in the model's training corpus, although one can clearly see that the mean $\log_e$ frequency increases when only considering non-OOV substitutions compared to all word substitutions.

Clean	A clever blend of fact and fiction.
GENETIC	A clever blend of **fait** [***fact***] and fiction. (0.00, 4.83)
PWWS	A **ingenious** [***clever***] **blending** [***blend***] of fact and **fabrication** [***fiction***]. (3.56, 5.55, 0.00, 3.81, 0.00, 4.39)

Figure 1: Illustration of the word frequency differences between the words selected for replacement (bold, italic and red) and their corresponding substitutions (bold and black) based on sequences crafted with the GENETIC and PWWS attacks against BERT_{base} on SST-2. The values above the highlighted words represent their $\log_e$ frequencies.

Figure 1 shows two adversarial sequences generated with the GENETIC and PWWS attacks against BERT_{base} on SST-2, and highlights the differences in word frequency between the replaced words and their corresponding substitutions.

4.2 Are classifiers vulnerable to low-frequency attacks?

We observe in Section 4.1 that all of the investigated adversarial attacks tend to replace words with less frequent substitutions. Nevertheless, it is worth noting that these findings do not directly show that the classification models are generally vulnerable to low-frequency words, since the attacks only implicitly utilize low-frequency word substitutions instead of explicitly searching for them. We hence investigate whether the three neural architectures are generally vulnerable to low-frequency word substitutions. To do this, we attack all three trained models with an additional adversarial attack that explicitly replaces selected input words with less frequent substitutions. Our proposed algorithm randomly selects a word x_i of an input sequence X and computes a set of substitution candidates $\mathcal{S}(x_i)$ that is defined by the union of the word's nearest neighbors in a pre-trained embedding space and its WORDNET synonyms (see Appendix C for details).

We then select a substitution x_i' for x_i by identifying the candidate substitution in $\mathcal{S}(x_i)$ exhibiting

the lowest $\log_e$ occurrence frequency with respect to the model's training corpus. We implement two variations of our attack. The first, denoted FREQUENCY$_r$, randomly selects words from an input sequence and replaces them as mentioned above. The second, denoted FREQUENCY$_p$, only accepts an individual word replacement if the prediction confidence placed on the sequence's true label decreases after the candidate word has been replaced.

Dataset	Classifier	FREQUENCY$_r$	FREQUENCY$_p$
IMDb	CNN	21.96	81.02
	LSTM	26.67	84.77
	BERT$_{base}$	17.15	54.63
SST-2	CNN	15.29	35.60
	LSTM	14.30	31.79
	BERT$_{base}$	11.50	21.31

Table 3: Attack success rates (%) of the low-frequency attack.

We adhere to previous experiments by allowing 20% of word changes made by the attack. The attack success rates of both attack variations can be found in Table 3. We observe that while FREQUENCY$_r$ exhibits poor attack performances, FREQUENCY$_p$ achieves to misclassify the majority of sequences for the IMDb dataset as well as an increased amount of sequences on SST-2. These findings indicate that while all three models seem to be robust against random low-frequency substitutions, adding a simple selection heuristic for more impactful word replacements yields strong attack performance increases. This clearly shows that, although the word frequency differences exist across a variety of attacks, relying on this heuristic alone does not suffice to confidently lead the investigated classification models into making false predictions.

5 Detecting textual adversarial examples

The observation of consistent word frequency differences between replaced words and their respective substitutions provides us with a simple way of detecting adversarial input manipulations. Specifically, we argue that the effects of adversarial word substitutions can be mitigated by conducting simple frequency-based transformations. Such transformations identify adversarial input tokens based on their low-frequency values and replace them with more frequent, semantically related tokens, to prevent the classification models from making false predictions caused by adversarial inputs.

5.1 Frequency-guided word substitutions

To do this, we propose *frequency-guided word substitutions* (FGWS), a detection method that exploits this idea to estimate whether a given textual sequence is an adversarial example. FGWS transforms a given sequence X into a sequence X' by replacing infrequent words with more frequent, semantically similar substitutions. Formally, for a given sequence X[1], we initially define the subset $X_E \subseteq X$ of words that are eligible for substitution as $X_E := \{x \in X \,|\, \phi(x) < \delta\}$, where $\delta \in \mathbb{R}_{>0}$ is a frequency threshold. FGWS then generates a sequence X' from X by replacing all eligible words with words that are semantically similar, but have higher occurrence frequencies in the model's training corpus. To do this, for each eligible word $x \in X_E$ we consider the set of replacement candidates $\mathcal{S}(x)$ and find a replacement x' by selecting $x' = \text{argmax}_{w \in \mathcal{S}(x)}\ \phi(w)$. Once we have identified all possible replacements for the words in X_E, we generate the sequence X' by replacing each eligible word x with x' if $\phi(x') > \phi(x)$. Given the sequences X and X', we assess to what extent the predictions made by the classification models are affected by the described procedure. A sequence is then considered an adversarial example if the prediction confidence of the given classifier decreases significantly after transformation. We propose both a continuous and a discrete discrimination method to measure this significance.

Discrete detection. In the discrete case, we simply assess whether the classifier changed its class prediction after transforming a given sequence. A sequence X is hence considered adversarial if $f^*(X) \neq f^*(X')$.

Continuous detection. The continuous case, in contrast, compares the absolute difference in prediction confidences between X and X'. We therefore first compute the prediction label $y = f^*(X)$ for X and define a threshold $\gamma \in [0, 1]$. The sequence X is then considered adversarial if $|\text{softmax}(f(X))_y - \text{softmax}(f(X'))_y| > \gamma$, i.e. if the absolute difference of probability mass placed on class y with respect to both the original and transformed sequences exceeds the threshold γ. The introduction of such a threshold allows to carefully control for the amount of false positives (i.e.

[1]For notational purposes, we here represent the sequence $X = x_1 x_2 \ldots x_n$ as a set $X = \{x_1, x_2, \ldots, x_n\}$.

Dataset	Classifier	Attack	AUC	Continuous			Discrete		
				TPR	FPR	F_1	TPR	FPR	F_1
IMDb	CNN	RANDOM	90.2	57.1 (30.3)	5.9 (4.2)	70.1 (45.0)	65.5	5.0	76.8
		PRIORITIZED	94.1	73.0 (49.0)	6.1 (2.0)	81.5 (64.9)	79.4	3.5	86.8
		GENETIC	93.8	71.7 (45.3)	6.3 (2.1)	80.5 (61.4)	76.5	3.6	84.9
		PWWS	94.0	72.6 (51.0)	6.1 (2.1)	81.3 (66.6)	77.8	3.6	85.8
	LSTM	RANDOM	80.9	61.1 (49.6)	9.7 (6.2)	71.5 (63.6)	46.9	0.0	63.9
		PRIORITIZED	88.1	72.6 (61.2)	10.2 (4.9)	79.5 (73.7)	59.5	0.6	74.3
		GENETIC	82.1	62.6 (50.9)	9.9 (4.9)	72.6 (65.3)	44.7	0.5	61.6
		PWWS	81.4	65.9 (57.6)	10.2 (4.9)	74.8 (70.9)	53.3	0.6	69.2
	BERT_{base}	RANDOM	97.0	89.9 (75.8)	8.1 (5.1)	90.8 (83.8)	56.6	1.0	71.8
		PRIORITIZED	97.2	92.8 (86.1)	7.0 (3.8)	92.9 (90.7)	75.1	1.4	85.1
		GENETIC	97.5	94.1 (86.4)	6.9 (3.8)	93.7 (90.8)	75.8	1.5	85.6
		PWWS	95.3	88.1 (77.3)	7.1 (4.0)	90.3 (85.3)	63.8	1.3	77.3
SST-2	CNN	RANDOM	86.3	41.5 (12.8)	6.4 (4.3)	56.1 (21.8)	59.6	7.4	71.3
		PRIORITIZED	88.2	50.3 (19.8)	5.1 (2.3)	64.7 (32.4)	63.4	7.5	74.2
		GENETIC	83.7	46.8 (27.4)	6.7 (3.5)	60.9 (41.9)	55.2	8.8	67.3
		PWWS	85.1	49.9 (21.7)	6.4 (3.0)	63.9 (34.8)	61.5	8.7	72.3
	LSTM	RANDOM	80.3	34.8 (15.7)	10.8 (4.5)	48.1 (26.2)	50.6	11.2	62.5
		PRIORITIZED	85.1	38.9 (16.2)	9.2 (4.4)	52.5 (26.8)	64.8	12.4	73.1
		GENETIC	84.2	46.8 (26.8)	8.4 (4.0)	60.3 (40.9)	60.9	11.3	70.7
		PWWS	83.3	38.6 (16.4)	8.3 (4.0)	52.5 (27.2)	61.0	11.1	70.9
	BERT_{base}	RANDOM	86.1	52.8 (25.0)	8.3 (4.2)	65.5 (38.7)	55.6	9.7	67.2
		PRIORITIZED	89.0	63.2 (21.2)	8.1 (4.2)	73.8 (33.9)	68.3	7.8	77.6
		GENETIC	85.2	53.0 (21.5)	8.5 (5.1)	65.6 (34.0)	55.8	8.1	68.1
		PWWS	86.2	58.1 (23.6)	8.6 (4.9)	69.7 (36.7)	62.8	8.0	73.5

Table 4: Performance results of FGWS. For the continuous detection, the reported rates are shown when γ is tuned to allow for 10% and 5% of false positives on the validation set (results for 5% in parentheses). TPR and FPR denote true and false positive rates, respectively.

unperturbed sequences that are identified as adversarial) detected by our method.

5.2 Comparisons and baselines

As textual adversarial example detection is a relatively new task, to the best of our knowledge the only existing approach to this task is the recently introduced DISP (*learning to discriminate perturbations*) framework (Zhou et al., 2019). Throughout the experiments, we compare FGWS to DISP and an additional baseline.

DISP. DISP is a perturbation discrimination approach that uses two independent components, a perturbation discriminator and an embedding estimator for token recovery, to identify individually perturbed tokens of a sequence and to reconstruct the replaced tokens. Both components are based on pre-trained BERT models to identify perturbed and reconstruct original tokens, respectively. DISP is trained on both character- and word-level perturbations. We adapt this framework to our task and train two DISP models for the IMDb and SST-2 datasets, respectively. For each dataset, we train both the discriminator and generator independently for 25 epochs on the training sets, and validate the trained checkpoints on the validation sets to identify a performance-maximizing combination of both components[2]. To do this, we use adversarial examples crafted with the RANDOM attack on the validation set, since it most closely follows the attacks utilized by Zhou et al. (2019). Once trained and evaluated, we employ the DISP modules to reconstruct the clean sequences from the adversarial ones perturbed with the four analyzed attacks. We then use the discrete detection technique as introduced for FGWS to detect adversarial sequences.

NWS. We furthermore introduce the *naive word substitutions* (NWS) baseline for better compar-

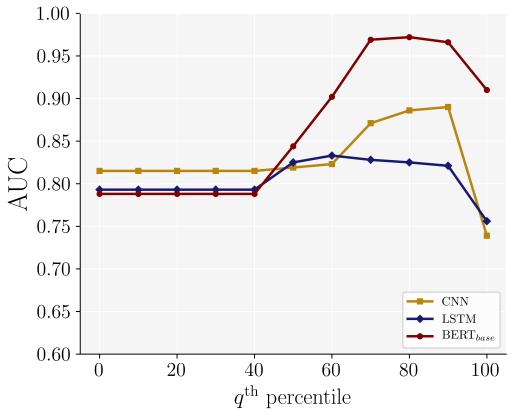
[2]We used the code made available at https://github.com/joey1993/bert-defender/.

isons between the different methods. For a given input sequence, NWS selects all out-of-vocabulary tokens in that sequence and, if possible, replaces each of the selected words with a randomly chosen word from a set of semantically related words. We restrict NWS to only allow word substitutions for which the replacement word occurs in the model's training vocabulary. In accordance to DISP, we use NWS with the discrete detection method in our experiments.

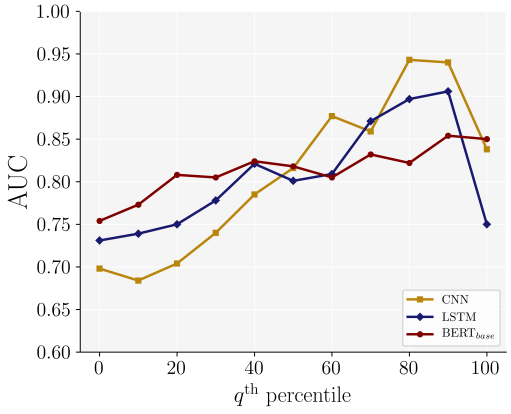
5.3 Experiments

We conduct a series of experiments by applying the detection methods to the adversarial sequences crafted by the introduced attacks on the subsets of both the IMDb and SST-2 datasets as explained in Section 3.5. We restrict the eligible words for replacement to non-stopwords, and tune the frequency threshold δ for each classifier-dataset combination on the validation set. To do this, we utilize the RANDOM attack to craft adversarial examples from all sequences of the validation set and compare our method's detection performance with different values for δ. Specifically, we set δ equal to the $\log_e$ frequency representing the q^{th} percentile of all $\log_e$ frequencies observed by the words eligible for replacement in the training set and experiment with $q \in \{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.

Moreover, we determine the threshold γ to take a value that approximates a limited number of false positive predictions made by the detection algorithm on the respective dataset's validation set. We select the threshold γ such that only up to 10% and 5% of the unperturbed sequences in the validation set are labeled as adversarial. For each word $x \in X_E$, we define the set of replacement candidates as the union $\mathcal{S}(x) = \mathcal{S}_E(x) \cup \mathcal{S}_W(x)$ of the word's K nearest neighbors in a pre-trained GLOVE word embedding space, denoted $\mathcal{S}_E(x)$, and its synonyms in WORDNET, denoted $\mathcal{S}_W(x)$. Here, we tune K on the validation set by setting it equal to the average number of WORDNET synonyms available for each word occurring in the validation set (yielding $K = 11$ for IMDb and $K = 16$ for SST-2), to approximate a balance between synonyms and embedding-based nearest neighbors in $\mathcal{S}(x)$. For NWS, we compute the set of semantically related words for each selected candidate analogously.



(a) IMDb



(b) SST-2

Figure 2: AUC performance scores of FGWS against the RANDOM attack on the validation sets with different values for δ. The x-axis shows the selected q^{th} percentiles of the $\log_e$ frequencies in the training corpus. The y-axis denotes the AUC score when δ is set to the $\log_e$ frequency value representing the specific q^{th} percentile.

5.4 Results

We report the experimental performance results of FGWS in both the discrete and continuous variations in Table 4. Here, the area under the receiver operating characteristic curve (AUC) is computed by interpreting the absolute difference in prediction confidence before and after transformation as the probability that a given sequence is an adversarial example. For both the discrete and continuous detection methods, the true positive rate (TPR) represents the percentage of perturbed sequences that FGWS correctly identifies as such and the false positive rate (FPR) denotes the percentage of unperturbed sequences that were identified as adversarial examples. The results show that the proposed method exhibits high AUC scores across

Dataset	Attack	Adv. acc.	Restored acc.			True positive rate			False positive rate			F_1		
			NWS	DISP	FGWS	NWS	DISP	FGWS	NWS	DISP	FGWS	NWS	DISP	FGWS
IMDb	RANDOM	86.3	87.4	88.1	**89.6**	30.3	40.4	**56.6**	**0.0**	2.0	1.0	46.5	56.7	**71.8**
	PRIORITIZED	26.0	54.7	64.8	**76.0**	44.4	59.7	**75.1**	**0.2**	1.5	1.4	61.4	74.0	**85.1**
	GENETIC	27.4	42.3	68.6	**77.0**	23.3	64.2	**75.8**	**0.2**	1.5	1.5	37.7	77.5	**85.6**
	PWWS	35.3	51.5	61.5	**71.6**	29.6	47.4	**63.8**	**0.2**	1.5	1.3	45.7	63.7	**77.3**
SST-2	RANDOM	88.2	**88.8**	86.0	85.9	34.7	43.1	**55.6**	**2.8**	9.7	9.7	50.5	56.4	**67.2**
	PRIORITIZED	60.8	74.3	73.4	**79.6**	45.5	51.7	**68.3**	**1.8**	6.4	7.8	61.8	65.4	**77.6**
	GENETIC	34.0	49.8	60.4	**66.8**	27.9	49.4	**55.8**	**1.2**	5.8	8.1	43.2	63.7	**68.1**
	PWWS	52.8	66.1	66.8	**75.1**	35.5	46.3	**62.8**	**1.4**	6.6	8.0	51.9	60.5	**73.5**

Table 5: Adversarial example detection performances for NWS (baseline), DISP and FGWS (discrete detection) when evaluated on attacks against BERT$_{base}$. Adv. acc shows the model's classification accuracy on the perturbed sequences. Restored acc. denotes the accuracies on the perturbed sequences after transformation. Underlined values in bold represent best scores per metric, dataset, attack and detection method.

Clean	Barney has created a tour de force that is weird wacky and wonderful.	*positive* (99.98%)
GENETIC	Barney has created a tour de force that is weird $^{0.00}$**loony** [***wacky***] and $^{2.48}$**resplendent** [***wonderful***].	*negative* (93.28%)
NWS	Barney has created a tour de force that is weird $^{2.20}$**weirdo** [***loony***] and $^{0.00}$**splendid** [***resplendent***].	*positive* (99.98%)
DISP	Barney has created a tour de force that is weird $^{0.00}$**,** [***loony***] and $^{0.00}$**too** [***resplendent***].	*positive* (62.11%)
FGWS	$^{4.13}$**Murphy** [***Barney***] has created a $^{3.14}$**world** [***tour***] de force that is weird $^{6.17}$**crazy** [***loony***] and $^{3.14}$**splendid** [***resplendent***].	*positive* (99.98%)
Clean	Thurman and lewis are hilarious throughout.	*positive* (99.98%)
GENETIC	Thurman and lewis are $^{3.22}$**droll** [***hilarious***] throughout.	*negative* (99.97%)
NWS	Thurman and lewis are droll throughout.	*negative* (99.97%)
DISP	Thurman and lewis are $^{3.89}$**fantastic** [***droll***] throughout.	*positive* (99.98%)
FGWS	$^{4.26}$**Robert** [***Thurman***] and $^{4.20}$**allen** [***lewis***] are $^{5.50}$**amusing** [***droll***] throughout.	*positive* (99.98%)

Figure 3: The three detection methods applied to adversarial examples created with the GENETIC attack for the BERT$_{base}$ classifier on SST-2. Clean represents an unperturbed sequence, GENETIC is the perturbed sequence generated by the attack. NWS, DISP and FGWS describe the sequences resulting from the respective substitution algorithms. The words highlighted in bold, italic and red represent the words selected for replacement by the attack and the detection mechanisms, the ones in bold and black denote the corresponding word substitutions. The values above the highlighted words denote their $\log_e$ frequencies.

all experiments, demonstrating the method's ability to accurately discriminate between unperturbed and adversarial sequences. Moreover, FGWS detects adversarial sequences accurately in multiple cases with both the discrete and continuous detection methods, exhibiting true positive rates of up to 94.1% on attacks against BERT$_{base}$ while at the same time predicting less than 7% false positives. However, one can clearly see the trade-off between the detection of true and false positives: when allowing for 10% of false positives on the validation set, FGWS performs consistently better in detecting true positives than when allowing for 5% of false positives. Nevertheless, even in the latter case our method detects a notable amount of adversarial sequences in the majority of our experiments. This indicates that the exploitation of the word frequency differences between unperturbed and perturbed sequences has the potential to detect a useful fraction of textual adversarial examples without creating an excessive burden of false positives.

Figure 2 illustrates the AUC scores of FGWS against the RANDOM attack on the validation sets with different values of δ (which were used to tune δ for testing). We observe that selecting higher values for δ, and therefore allowing FGWS to manipulate tokens with higher occurrence frequencies, is beneficial for the detection performance. Nevertheless, we also observe notable decreases for the 100^{th} percentile, indicating that allowing FGWS to substitute the most frequent words can have a crucial impact on detection performance.

Comparison to NWS and DISP. The comparison of FGWS to both NWS and DISP can be found in Table 5. The reference model used for the comparison is BERT$_{base}$, in accordance to the evaluations as presented by Zhou et al. (2019). We utilize the discrete detection method on the sequences manipulated by each algorithm. The column Adv. acc. denotes the adversarial classification accuracy

of BERT$_{base}$ on the perturbed sequences, and Restored acc. represents the model's accuracy on the adversarial sequences after transformation with the three detection methods. We observe that FGWS best restores the model's original classification accuracy for the majority of the comparisons, thereby showing to be effective in mitigating the effects caused by the individual attacks (the accuracies on the clean test data can be found in Table 1). While DISP outperforms NWS in terms of true positive rates and F_1 across all experiments, we can see that FGWS consistently outperforms both methods for the same comparisons. These results suggest that *i)* simply mapping OOV tokens to semantically similar vocabulary tokens (NWS) represents an effective detection baseline, *ii)* utilizing contextualized representations (DISP) improves upon this baseline approach, showing that adversarial word substitutions are identifiable through contextual information, and *iii)* relying solely on frequency-guided substitutions without incorporating contextual information (FGWS) shows to be most effective.

Moreover, the direct comparison between NWS and FGWS again underlines the importance of utilizing word frequencies as guidance for the word substitutions: while NWS is not guided by word frequency characteristics to perform the word replacements, we observe that FGWS outperforms NWS by a large margin in terms of F_1, demonstrating the effectiveness of mapping infrequent words to semantically similar, more frequent words in order to detect textual adversarial examples.

Figure 3 provides two examples of adversarial sequences generated with the GENETIC attack and the three corresponding transformed sequences using NWS, DISP and FGWS. The GENETIC attack achieves to generate adversarial examples by replacing multiple words in each sequence. The detection methods, however, identify parts of the adversarial substitutions and replace them with different, semantically similar words. The resulting transformed sequences are then again correctly classified (except NWS in the second example).

5.5 Limitations

It is worth mentioning that compared to FGWS, DISP represents a more general perturbation discrimination approach since it is trained to detect both character- and word-level adversarial perturbations, whereas FGWS solely focuses on word-level attacks. Furthermore, and in contrast to our work, DISP is evaluated on simple word substitution (comparable to our RANDOM and PRIORITIZED attack baselines) and character manipulation attacks. Since the present work focuses more generally on the word frequency properties of textual adversarial examples, we decided to include more sophisticated adversarial attacks (PWWS and GENETIC) to *i)* demonstrate that the observed frequency characteristics hold across different classes of attacks and *ii)* since we believe that such attacks have a stronger relation to the practical implications of being able to detect textual adversarial examples.

Dataset	**Classifier**	**Clean**	FGWS$_{test}$	DISP$_{test}$
IMDb	CNN	86.63	86.16	84.87
	LSTM	86.61	86.67	85.80
	BERT$_{base}$	90.84	90.87	90.12
SST-2	CNN	84.29	80.78	83.03
	LSTM	83.86	79.63	83.58
	BERT$_{base}$	92.20	87.53	89.62

Table 6: Classification accuracies before and after applying FGWS and DISP to the clean test sets.

5.6 FGWS and classification performance

While FGWS shows to aid in detecting adversarial sequences, such transformations might still result in semantic shifts of the manipulated sequences and hence a decrease in model classification performance on unperturbed sequences after transformation. We explicitly investigate this by transforming the sequences in the test sets using FGWS (with δ set to the same values as for the detection task), and evaluate the models' classification accuracies on the test sets after transformation. Table 6 shows the classification accuracies for all three models when tested on the sequences transformed with FGWS (denoted FGWS$_{test}$). For comparison, we also report the accuracies after transforming the test sets using DISP (denoted DISP$_{test}$). Here, one can observe that FGWS has only little influence on classification performance on the IMDb dataset and even leads to slight improvements, whereas slight decreases are observed on DISP$_{test}$. On SST-2, notable performance decreases can be observed with respect to both DISP and FGWS, although the decreases are more dominant on the data transformed using FGWS. This indicates that an increased ability to detect adversarial examples might lead to performance decreases on unperturbed data (see Appendix D for a trade-off comparison).

6 Conclusion

We have shown that the word occurrence frequency characteristics of adversarial word substitutions can be leveraged effectively to discriminate between unperturbed and perturbed sequences in the context of adversarial attacks against neural text classification models. Our proposed approach outperforms existing adversarial example detection methods despite representing a much simpler approach to this task. In future work, we aim to further utilize the demonstrated frequency characteristics to increase the robustness of text processing models against adversarial attacks, and to exploit whether word frequency characteristics can be leveraged as effectively across other natural language processing tasks in adversarial settings.

Acknowledgements

This research was supported by the Dawes Centre for Future Crime at University College London (UCL). We would like to thank Max Bartolo as well as the Natural Language Processing and Computational Security Science research groups at UCL's Department of Computer Science for their helpful discussions and feedback.

References

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Melika Behjati, Seyed-Mohsen Moosavi-Dezfooli, Mahdieh Soleymani Baghshah, and Pascal Frossard. 2019. Universal adversarial attacks on text classifiers. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7345–7349. IEEE.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. Technical report, Google.

Yong Cheng, Lu Jiang, and Wolfgang Macherey. 2019. Robust neural machine translation with doubly adversarial inputs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4324–4333, Florence, Italy. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. 2019. Text processing like humans do: Visually attacking and shielding NLP systems. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1634–1647, Minneapolis, Minnesota. Association for Computational Linguistics.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops (SPW)*, pages 50–56.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Melbourne, Australia. Association for Computational Linguistics.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Po-Sen Huang, Robert Stanforth, Johannes Welbl, Chris Dyer, Dani Yogatama, Sven Gowal, Krishnamurthy Dvijotham, and Pushmeet Kohli. 2019. Achieving verified robustness to symbol substitutions via interval bound propagation. In *Proceedings of the 2019 Conference on Empirical Methods*

in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4081–4091, Hong Kong, China. Association for Computational Linguistics.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4120–4133, Hong Kong, China. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial examples in the physical world. *ICLR Workshop*.

Qi Lei, Lingfei Wu, Pin-Yu Chen, Alexandros G Dimakis, Inderjit S Dhillon, and Michael Witbrock. 2019. Discrete adversarial attacks and submodular optimization with applications to text classification. In *SysML 2019*.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2018. Textbugger: Generating adversarial text against real-world applications. *arXiv preprint arXiv:1812.05271*.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016a. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016b. Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

Yitong Li, Trevor Cohn, and Timothy Baldwin. 2017. Robust training under linguistic adversity. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 21–27, Valencia, Spain. Association for Computational Linguistics.

Bin Liang, Hongcheng Li, Miaoqiang Su, Pan Bian, Xirong Li, and Wenchang Shi. 2018. Deep text classification can be fooled. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, pages 4208–4215. AAAI Press.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148, San Diego, California. Association for Computational Linguistics.

Nicolas Papernot, Patrick Drew McDaniel, Ananthram Swami, and Richard Harang. 2016. Crafting adversarial input sequences for recurrent neural networks. In *MILCOM 2016 - 2016 IEEE Military Communications Conference*, Proceedings - IEEE Military Communications Conference MILCOM, pages 49–54, United States. Institute of Electrical and Electronics Engineers Inc.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, and Zachary C. Lipton. 2019. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th*

Annual Meeting of the Association for Computational Linguistics, pages 5582–5591, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Yi-Ting Tsai, Min-Chu Yang, and Han-Yu Chen. 2019. Adversarial attack on sentiment classification. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 233–240, Florence, Italy. Association for Computational Linguistics.

Yicheng Wang and Mohit Bansal. 2018. Robust machine comprehension models via adversarial training. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 575–581, New Orleans, Louisiana. Association for Computational Linguistics.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh, Jane-Ling Wang, and Michael I. Jordan. 2020. Greedy attack and gumbel attack: Generating adversarial examples for discrete data. *Journal of Machine Learning Research*, 21(43):1–36.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.

Yichao Zhou, Jyun-Yu Jiang, Kai-Wei Chang, and Wei Wang. 2019. Learning to discriminate perturbations for blocking adversarial attacks in text classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4903–4912, Hong Kong, China. Association for Computational Linguistics.

A Model architectures, hyperparameters and training details

A.1 CNN

The CNN architecture consists of $L = 3$ convolutional layers with kernel sizes 2, 3 and 4 and $F = 100$ feature maps for each convolutional layer. The CNN's penultimate layer applies max-pooling over time to produce an $L \cdot F$ dimensional output representation which is then projected to a C-dimensional class logit representation.

A.2 LSTM

We utilize a single-layer unidirectional LSTM with a hidden state size of 128. The LSTM's initial hidden and cell states are each initialized with the 128-dimensional zero vector. The final layer consists of an affine transformation projecting the mean of the output states from each time step to a C-dimensional logit representation.

A.3 BERT$_{base}$

Our BERT-based classification model utilizes a pretrained BERT$_{base}$ model provided by the *Hugging Face Transformers* library (Wolf et al., 2019).

A.4 Training details

Both the LSTM and the CNN use *Dropout* (Srivastava et al., 2014) during training with a rate of 0.1 before applying the output layer. We trained all three models for 20 epochs using the *Adam* optimizer (Kingma and Ba, 2014).

The CNN and LSTM models were trained with batch size 100 and a learning rate of $1 \cdot 10^{-3}$, BERT_{base} was trained with batch size 32 and a learning rate of $2 \cdot 10^{-5}$. We used early stopping for all three models by validating model performance on the validation set after each epoch.

We furthermore did not filter the training vocabularies for both datasets by imposing a maximum vocabulary size. Hence, the IMDb training set generates a vocabulary comprising 64,824 words, and processing all training sequences from SST-2 yields a vocabulary size of 13,845 words.

B Attack implementation details

B.1 Random, Prioritized and PWWS

All three attack implementations are based on the code as provided by Ren et al. (2019) on GitHub[3]. We follow the authors' implementation of the PWWS attack by only selecting conjunctions, adjectives, nouns, adverbs and verbs for replacement. We also follow their restriction that synonym replacements must be at least three characters long and must have the same part-of-speech tag as the selected word. We keep these constraints for the implementations of the RANDOM and PRIORITIZED baselines as well.

B.2 PWWS

In their proposed attack algorithm, Ren et al. (2019) compute the most frequently occurring named entities for each class across all sequences occurring in each dataset. It is worth noting that when computing such named entities for the IMDb dataset (the only dataset that is used in both Ren et al. (2019)'s and our experiments), we obtain different results as compared to the ones as provided by the authors. However, this has no notable effect on the attack performances, since our reimplementation of the attack is highly effective with attack performances comparable to those reported for the original implementation (see Table 1).

[3] https://github.com/JHL-HUST/PWWS

B.3 Genetic

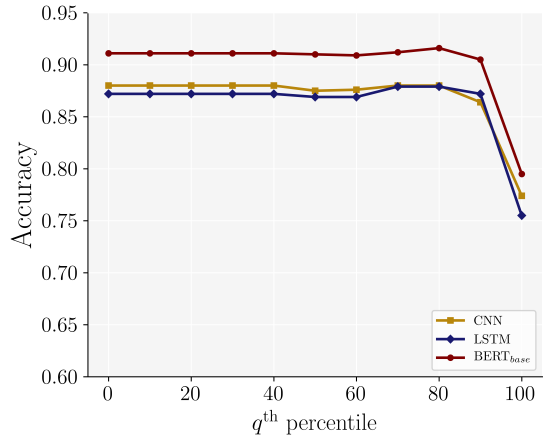
Note that we utilize a different language model for the `Perturb` subroutine as compared to the original implementation by Alzantot et al. (2018). While Alzantot et al. (2018) employ the Google 1 billion words language model (Chelba et al., 2013), we instead utilize the recently proposed GPT-2 language model (Radford et al., 2019) and compute the sequences' perplexity scores using the exponentialized language modelling loss (we employ the pre-trained `GPT2LMHeadModel` language model from Wolf et al. (2019)). We compute the perplexity scores for each perturbed sequence only around the respective replacement words by only considering a subsequence ranging from the 5 words before to the 5 words after an inserted replacement. The motivation for using a different language model as compared to the original implementation is due to computational complexity reasons, since we observed a notable decrease in attack runtime with our modification. All other parameters of the attack (e.g. the number of generations and population size) are directly adapted from Alzantot et al. (2018).

We furthermore restrict the words eligible for replacement by the GENETIC attack to those that are at least three characters long and are neither stopwords nor the end-of-sentence token. Since the attack computes nearest neighbors for a selected word from a pre-trained embedding space, we furthermore can only select words for which there exists an embedding representation in this pre-trained space.

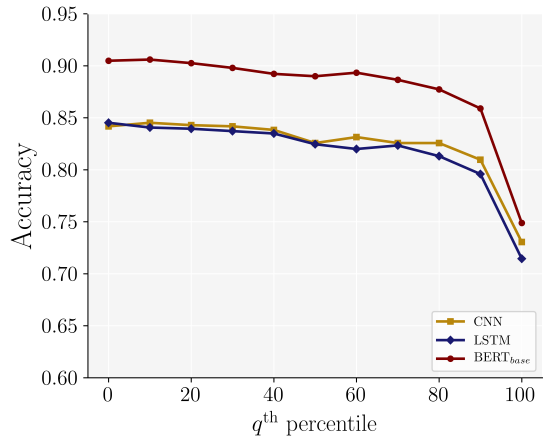
C Details of the low-frequency attack

For both variations of the FREQUENCY attack, we identify the set of substitution candidates for each replaced word as follows: for the word embeddings, we adhere to Alzantot et al. (2018) by utilizing a set of 300-dimensional PARAGRAM vectors (Wieting et al., 2015) trained using the counter-fitting method as introduced by Mrkšić et al. (2016) to identify a word's K nearest neighbors. This method is used to ensure that the queried nearest neighbors are synonyms of the replacement candidate. We use Euclidean distance to compute an embedding's nearest neighbors.

To ensure a balanced combination of lexical and embedding-based replacement candidates, we set the number of nearest neighbors in embedding



(a) IMDb



(b) SST-2

Figure 4: Classification accuracies on the validation sets with different values for δ. The x-axis shows the selected q^{th} percentiles of the $\log_e$ frequencies in the training corpus. The y-axis denotes the accuracy when δ is set to the $\log_e$ frequency value representing the specific q^{th} percentile.

space considered for each word equal to the average amount of WORDNET synonyms of all words in the test set (yielding $K = 15$ for SST-2 and $K = 11$ for IMDb). We choose both embeddings- and lexicon-based synonyms to include substitution candidates that were used in both the GENETIC (Alzantot et al., 2018) and PWWS (Ren et al., 2019) attacks.

D Varying δ thresholds and classification performance

We investigate the impact of varying δ thresholds by analyzing the change in classification performance on the validation sets after applying FGWS. Figure 4 shows the model accuracies on the validation sets of both datasets with different values for δ. Here, δ is set to represent the $\log_e$ frequency at the q^{th} percentile of all $\log_e$ frequencies in the training corpus, where we experiment with $q \in \{0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. We observe a tendency towards decreasing classification performance with increasing values of δ for the SST-2 dataset. For the IMDb dataset, the classification performance remains unaffected up to the 40^{th} percentile for all three classifiers, and then fluctuates slightly before it decreases drastically at the 100^{th} percentile.

When optimizing δ for maximum classification accuracy on the validation set of each dataset, δ is optimized at the 0^{th} percentile for both the CNN on IMDb and the LSTM on SST-2. For both the CNN and BERT$_{base}$ on SST-2, δ is optimized at the 10^{th} percentile. For LSTM on IMDb, δ represents the 70^{th} percentile, and for BERT$_{base}$ on IMDb it represents the 80^{th} percentile.

Analyzing these findings in light of the results as shown in Figure 2, we can clearly observe a trade-off between classification accuracy and adversarial sequence detection performance when choosing different values for δ.