A survey on Adversarial Attacks and Defenses in Text

Wenqi Wang, Lina Wang*, Member, IEEE, Benxiao Tang, Run Wang, Aoshuang Ye

Abstract—Deep neural networks (DNNs) have shown an inherent vulnerability to adversarial examples which are maliciously crafted on real examples by attackers, aiming at making target DNNs misbehave. The threats of adversarial examples are widely existed in image, voice, speech, and text recognition and classification. Inspired by the previous work, researches on adversarial attacks and defenses in text domain develop rapidly. In order to make people have a general understanding about the field, this article presents a comprehensive review on adversarial examples in text, including attack and defense approaches. We analyze the advantages and shortcomings of recent adversarial examples generation methods and elaborate the efficiency and limitations on the countermeasures. Finally, we discuss the challenges in adversarial texts and provide a research direction of this aspect.

Index Terms—Adversarial attacks and defenses, Adversarial examples, Deep neural networks, Text domain, Testing and verification.

1 INTRODUCTION

Nowadays, DNNs have solved masses of significant practical problems in various areas like computer vision [1], [2], audio [3], [4], natural language processing (NLP) [5], [6] etc. Due to the great success, systems based on DNN are widely deployed in physical world, including some sensitive security tasks. However, Szegedy et al. [7] found an interesting fact that a crafted input with small perturbations could easily fool DNN models. This kind of inputs is called adversarial examples. Certainly, with the development of theory and practice, the definitions of adversarial examples [7], [8], [9], [10] are varied. But these definitions have two cores in common. One is that the perturbations are small and the ability of fooling DNN models is the other. It naturally raises a question why adversarial examples exist in DNNs. The reason why they are vulnerable to adversarial examples is probably because of DNNs linear nature. Goodfellow et al. [8] then gave this explanation after adversarial examples arose. Researchers therefore treat adversarial examples as a security problem and pay much attention to works of adversarial attacks and defenses [11], [12].

In recent years, category of adversarial examples becomes diverse, varying from image to audio and others. That means almost all deployed systems based on DNN are under the potential threat of adversarial attacks. For example, sign recognition system [13], object recognition system [14], audio recognition or control system [15], [16], [17] and malware detection system [18], [19] are all hard to defend against this kind of attack. Of course, systems for NLP tasks are also under the threat of adversarial examples, like text classification, sentiment analysis, question answering system, recommendation system and so on.

Fig. 1: Example of attack on DNNs by DeepWordBug [24]: the generated adversarial example based on original one can fool DNN model make error classification from positive to negative. The recommendation score of this film will decrease if massive adversarial examples of this type are applied by competitors, resulting in a low box office.

In real life, people are increasingly inclined to search for related comments before shopping, eating or watching film and the corresponding items with recommendation score will be given at the same time. The higher the score is, the more likely it is to be accepted by humans. These recommendation apps mainly take advantage of sentiment analysis with others previous comments [20]. Thus attackers could generate adversarial examples based on natural comments to smear competitors (see Fig.1 for instance) or do malicious recommendations for shoddy goods with the purpose of profit or other malicious intents. Apart from mentioned above, adversarial examples can also poison network environment and hinder detection of malicious information [21], [22], [23]. Hence, it is significant to know how adversarial attacks conduct and what measures can defend against them to make DNNs more robust.

This paper presents a comprehensive survey on adversarial attacks and defenses in text domain to make interested readers have a better understanding of this concept. It presents the following contributions:
• We systematically analyze recent adversarial attack approaches in text. For each of them, we introduce its implementation procedure and give a brief review on the advantages and shortcomings.
• We summarize the representative metric methods of adversarial example in both image and text. Then we compare the differences between them and point out why metric methods in image can not be directly used in text.
• In order to defend against adversarial attack, we also review the existing defense methods applied in text. As another important means of defense, testing and verification against adversarial examples are described to readers too, which have not appeared in other surveys.

The remainder of this paper is organized as follows: we first give some background about adversarial examples in section 2. In section 3 we review the adversarial attacks for text classification and other real-world NLP tasks. The researches with the central topic of defense are introduced in section 4 and 5. One of them is on existing defense methods in text and the other is about how to improve the robustness of DNNs from another point of view. The discussion and conclusion of the article is in section 6 and 7.

2 BACKGROUND
In this section, we describe some research background on the textual adversarial examples, including representation of symbol and attack types and scenarios.

2.1 Adversarial Example Formulation
The function of a pre-trained text classification model $F$ is mapping from input set to the label set. For a clean text example $x$, it is correctly classified by $F$ to ground truth label $y \in Y$, where $Y$ including $\{1, 2, \ldots, k\}$ is a label set of $k$ classes. An attacker aims at adding small perturbations in $x$ to generate adversarial example $x'$, so that $F(x') = y' (y' \neq y)$. Generally speaking, a good $x'$ should not only be misclassified by $F$, but also imperceptible to humans, robust to transformations as well as resilient to existing defenses depending on the adversarial goals [25]. Hence, constraint conditions (e.g. semantic similarity, distance metric, etc.) are appended to make $x'$ be indistinguishable from $x$ in some works and exploit it to cause classification errors like Fig. 1.

Under this condition, adversaries generally train their own models and utilize the transferability [8], [26] of adversarial examples to carry out an attack.

**White-box.** In white-box attack, adversaries have full access to target model and they can know all about architectures, parameters and weights of the model. Certainly, both white-box and black-box attacks can not change the model and training data.

According to the purpose of the adversary, adversarial attacks can be categorized as targeted attack and non-targeted attack.

**Targeted attack.** In this case, the generated adversarial example $x'$ is purposeful classified as class $t$ which is the target of an adversary.

**Non-targeted attack.** In this case, the adversary only wants to fool the model and the result $y'$ can be any class except for ground truth $y$.

2.3 Metric
There exists an important issue that the generated adversarial texts should not only be able to fool target models, but also need to keep the perturbations imperceptible. In other words, good adversarial examples should convey the same semantic meaning with the original ones so that metric measures are required to ensure this case. We describe different kinds of measures to evaluate the utility of adversarial examples in image and text. Then we analyze the reasons why metric measures in image are not suitable in text.

2.3.1 Metric measures in image
In image, almost all recent studies on adversarial attacks adopt $L_p$ distance as a distance metric to quantify the imperceptibility and similarity of adversarial examples. The generalized term for $L_p$ distance is as follows:

$$\|\Delta x\|_p = \sqrt[p]{\sum_{i=1}^{n}|x'_i - x_i|^p}$$  \(1\)

where $\Delta x$ represents the perturbations. This equation is a definition of a set of distances where $p$ could be $0, 1, \infty$ and so on. Specially, $L_0$ [27], [28], [29], $L_2$ [29], [30], [31], [32] and $L_{\infty}$ [7], [8], [32], [33], [34], [35] are the three most frequently used norms in adversarial images.

• $L_0$ distance evaluates the number of changed pixels before and after modifications. It seems like edit distance, but it may not directly work in text. Because results of altered words in text are varied. Some of them are similar to original words and the others may be contrary, even though the $L_0$ distance of them is same.

• $L_2$ distance is the Euclidean distance. The original Euclidean distance is the beeline from one point to another in Euclidean space. As the mapping of image, text or others to it, Euclidean space becomes a metric space to calculate the similarity between two objects represented as the vector.

• $L_{\infty}$ distance measures the maximum change as follows:

$$\|\Delta x\|_{\infty} = \max(|x'_1 - x_1|, \ldots, |x'_n - x_n|)$$  \(2\)
Although $L_\infty$ distance is thought to be the optimal distance metric to use in some work, but it may fail in text. The altered words may not exist in pre-trained dictionary so that they are considered to be unknown words and their word vectors are also unknown. As a result, $L_\infty$ distance is hard to calculate.

There are also other metric measures(e.g. structural similarity [36], perturbation sensitivity [37]) which are typical methods for image. Some of them are considered to be more effective than $L_p$ distance, but they can not directly used too.

### 2.3.2 Metric measures in text

In order to overcome the metric problem in adversarial texts, some measures are presented and we describe five of them which have been demonstrated in the pertinent literature.

**Euclidean Distance.** In text, for two given word vectors $\vec{m} = (m_1, m_2, \ldots, m_k)$ and $\vec{n} = (n_1, n_2, \ldots, n_k)$, the Euclidean distance is:

$$D(\vec{m}, \vec{n}) = \sqrt{(m_1-n_1)^2 + \ldots + (m_k-n_k)^2} \quad (3)$$

Euclidean distance is more used for the metric of adversarial images [29], [30], [31], [32] than texts with a generalized term called $L_2$ norm or $L_\infty$ distance.

**Cosine Similarity.** Cosine similarity is also a computational method for semantic similarity based on word vector by the cosine value of the angle between two vectors. Compared with Euclidean distance, the cosine distance pays more attention to the difference in direction between two vectors. The more consistent the directions of two vectors are, the greater the similarity is. For two given word vectors $\vec{m}$ and $\vec{n}$, the cosine similarity is:

$$D(\vec{m}, \vec{n}) = \frac{\vec{m} \cdot \vec{n}}{||\vec{m}|| \cdot ||\vec{n}||} = \frac{\sum_{i=1}^{k} m_i \times n_i}{\sqrt{\sum_{i=1}^{k} (m_i)^2} \times \sqrt{\sum_{i=1}^{k} (n_i)^2}} \quad (4)$$

But the limitation is that the dimensions of word vectors must be the same.

**Jaccard Similarity Coefficient.** For two given sets A and B, their Jaccard similarity coefficient is:

$$J(A, B) = |A \cap B|/|A \cup B| \quad (5)$$

where $0 \leq J(A, B) \leq 1$. It means that the closer the value of $J(A, B)$ is to 1, the more similar they are. In the text, intersection $A \cap B$ refers to similar words in the examples and union $A \cup B$ is all words without duplication.

**Word Movers Distance (WMD).** WMD [38] is a variation of Earth Mover’s Distance (EMD) [39]. It can be used to measure the dissimilarity between two text documents, relying on the travelling distance from embedded words of one document to another. In other words, WMD can quantify the semantic similarity between texts. Meanwhile, Euclidean distance is also used in the calculation of WMD.

**Edit Distance.** Edit distance is a way to measure the minimum modifications by turning a string to another. The higher it is, the more dissimilar the two strings are. It can be applied to computational biology and natural language processing. Levenshtein distance [40] is also referred to as edit distance with insertion, deletion, replacement operations used in work of [24].

### 2.4 Datasets in Text

In order to make data more accessible to those who need it, we collect some datasets which have been applied to NLP tasks in recent literatures and a brief introductions are given at the same time. These data sets can be downloaded via the corresponding link in the footnote.

**AG’s News**[4]. This is a news set with more than one million articles gathered from over 2000 news sources by an academic news search engine named ComeToMyHead. The provided db version and xml version can be downloaded for any non-commercial use.

**DBPedia Ontology**[2]. It is a dataset with structured content from the information created in various Wikimedia projects. It has over 68 classes with 2795 different properties and now there are more than 4 million instances included in this dataset.

**Amazon Review**[6]. The Amazon review dataset has nearly 35 million reviews spanning Jun 1995 to March 2013, including product and user information, ratings, and a plaintext review. It is collected by over 6 million users in more than 2 million products and categorized into 33 classes with the size ranging from KB to GB.

**Yahoo! Answer**[7]. The corpus contains 4 million questions and their answers, which can be easily used in the question answer system. Besides that, a topic classification dataset is also able to be constructed with some main classes.

**Yelp Review**[8]. The provided data is made available by Yelp to enable researchers or students to develop academic projects. It contains 4.7 million user reviews with the type of json files and sql files.

**Movie Review (MR)**[9]. This is a labeled dataset with respect to sentiment polarity, subjective rating and sentences with subjectivity status or polarity. Probably because it is labeled by humans, the size of this dataset is smaller than others, with a maximum of dozens of MB.

**MPQA Opinion Corpus**[10]. The Multi-Perspective Question Answering (MPQA) Opinion Corpus is collected from a wide variety of news sources and annotated for opinions or other private states. Three different versions are available to people by the MITRE Corporation. The higher the version is, the richer the contents are.

**Internet Movie Database (IMDB)**[11]. IMDBs is crawled from Internet including 50000 positive and negative reviews and average length of the review is nearly 200 words. It is usually used for binary sentiment classification including richer data than other similar datasets. IMDB also contains the additional unlabeled data, raw text and already processed data.

**SNLI Corpus**[12]. The Stanford Natural Language Inference (SNLI) Corpus is a collection with manually labeled data mainly for natural language inference (NLI) task. There are nearly five hundred thousand sentence pairs written.

1. [http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html](http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html)
2. [https://wiki.dbpedia.org/services-resources/ontology](https://wiki.dbpedia.org/services-resources/ontology)
3. [http://snap.stanford.edu/data/web-Amazon.html](http://snap.stanford.edu/data/web-Amazon.html)
4. [https://sourceforge.net/projects/yahoodataset/](https://sourceforge.net/projects/yahoodataset/)
5. [https://www.yelp.com/dataset/download](https://www.yelp.com/dataset/download)
6. [http://www.cs.cornell.edu/people/pabo/movie-review-data/](http://www.cs.cornell.edu/people/pabo/movie-review-data/)
7. [http://mpqa.cs.pitt.edu/](http://mpqa.cs.pitt.edu/)
8. [http://ai.stanford.edu/amaas/data/sentiment/](http://ai.stanford.edu/amaas/data/sentiment/)
9. [https://nlp.stanford.edu/projects/snli/](https://nlp.stanford.edu/projects/snli/)
by humans in a grounded context. More details about this corpus can be seen in the research of Samuel et al. [31].

3 Adversarial Attacks in Text

Because the purpose of adversarial attacks is to make DNNs misbehave, they can be seen as a classification problem in a broad sense. And majority of recent representative adversarial attacks in text is related to classification so that we categorize them with this feature. In this section, we introduce the majority of existing adversarial attacks in text. Technical details and corresponding comments of each attack method described below are given to make them more clearly to readers.

3.1 Non-target attacks for classification

Adversarial attacks can be subdivided in many cases which are described in section 2.2. With the purpose of more granular division of classification tasks, we introduce these attack methods group by group based on the desire of attackers. In this part, studies below are all non-target attacks that attackers do not care the category of misclassified results.

3.1.1 Adversarial Input Sequences

Papernot et al. [42] might be the first to study the problem of adversarial example in text and contributed to producing adversarial input sequences on Recurrent Neural Network (RNN). They leveraged computational graph unfolding [43] to evaluate the forward derivative [27], i.e. Jacobian, with respect to embedding inputs of the word sequences. Then for each word of the input, fast gradient sign method (FGSM) [8] was used on Jacobian tensor evaluated above to find the perturbations. Meanwhile, in order to solving the mapping problem of modified word embedding, they set a special dictionary and chose words to replace the original ones. The constraint of substitution operation was that the word to replace the origin one. Thus, it improved the interpretability of adversarial examples by adversarial training. The difference was that three modification strategies of insertion, replacement and deletion were introduced by Samanta et al. [45] to generate adversarial examples. Meanwhile, in order to preserve the readability of these examples, edit distance was used by the authors.

3.1.4 Interpretable Adversarial Training Text(iAdv-Text)

Different from other methods, Sato et al. [45] operated in input embedding space for text and reconstructed adversarial examples to misclassify the target model. The core idea of this method was that they searched for the weights of the direction vectors which maximized loss functions with overall parameters W as follows:

$$\alpha_{\text{iAdvT}} = \arg\max_{\alpha} \left\{ \ell (\vec{w}, \sum_{k=1}^{\|V\|} a_k \vec{d}_k, \hat{Y}, W) \right\}$$ (7)

where $$\sum_{k=1}^{\|V\|} a_k \vec{d}_k$$ was the perturbation generated from each input on its word embedding vector $$\vec{w}$$ and $$\vec{d}$$ was the direction vector from one word to another in embedding space. Because $$N_{\text{iAdvT}}$$ in Eq. (7) was hard to calculate, the authors used Eq. (8) instead:

$$\alpha_{\text{iAdvT}} = \frac{\epsilon g}{\|g\|_2}, g = \nabla_{\alpha} \ell (\vec{w}, \sum_{k=1}^{\|V\|} a_k \vec{d}_k, \hat{Y}, W)$$ (8)

The loss function of iAdvT was then defined based on $$N_{\text{iAdvT}}$$ as an optimization problem by jointly minimizing objection functions on entire training dataset D as follows:

$$\hat{W} = \frac{1}{|D|} \arg\min_{W} \left\{ \sum_{(X, Y) \in D} \ell (\hat{X}, \hat{Y}, W) + \lambda \sum_{(X, Y) \in D} \ell (\hat{X}_{+\gamma(\alpha_{\text{iAdvT}})}(\hat{Y}), W) \right\}$$ (9)

Comparing with Miyato et al. [47], iAdv-Text restricted the direction of perturbations to find a substitute which was in the predefined vocabulary rather than an unknown word to replace the origin one. Thus, it improved the interpretability of adversarial examples by adversarial training. The authors also took advantage of cosine similarity to select a better perturbation at the same time.

Similarly, Gong et al. [50] also searched for adversarial perturbations in embedding space, but their method was gradient-based. Even though WMD was used by the authors to measure the similarity of clean examples and adversarial
examples, the readability of generated results seemed a little poor.

3.1.5 TextBugger

Li et al. [49] proposed an attack framework TextBugger for generating adversarial examples to trigger the deep learning-based text understanding system in both black-box and white-box settings. They followed the general steps to capture important words which were significant to the classification and then crafted on them. In white-box setting, the Jacobian matrix was used to calculate the importance of each word as follows:

\[ C_x = J_{F(i,y)} = \frac{\partial F_y(x)}{\partial x_i} \quad (10) \]

where \( F_y(\cdot) \) represented the confidence value of class \( y \). The slight changes of words were in character-level and word-level respectively by operations like insertion, deletion, swap and substitution. In black-box setting, the authors segmented documents into sequences and probed the target model to filter out sentences with different predicted labels from the original. The odd sequences were sorted in an inverse order by their confidence score. Then important words were calculated by removing method as follows:

\[ C_{x_i} = F_y(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n) - F_y(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \quad (11) \]

The last modification process was same as that in white-box setting.

3.2 Target attacks for classification

For target attack, attackers purposefully control the category of output to be what they want and the generated examples have similar semantic information with clean ones. This kind of attacks are described one by one in the following part.

3.2.1 Text-Fool

Different from works in [42], [45], Liang et al. [50] first demonstrated that FGSM could not be directly applied in text. Because input space of text is discrete, while image data is continuous. Continuous image has tolerance of tiny perturbations, but text does not have this kind of feature. Instead, the authors utilized FGSM to determine what, where and how to insert, remove and modify on text input. They conducted two kinds of attacks in different scenarios and used the natural language watermarking [51] technique to make generated adversarial examples compromise their utilities.

In white-box scenario, the authors defined the concepts of hot training phrases and hot sample phrases which were both obtained by leveraging the backpropagation algorithm to compute the cost gradients of samples. The former one shed light on what to insert and the later implied where to insert and modify. In black-box scenario, authors used the idea of fuzzing technique [52] for reference to obtain hot training phrases and hot sample phrases. One assumption was that the target model could be probed. Samples were fed to target model and then isometric whitespace was used to substitute origin word each time. The difference between two classification results was each word’s deviation. The larger it was, the more significant the corresponding word was to its classification. Hence, hot training phrases were the most frequent words in a set which consisted of the largest deviation word for each training sample. And hot sample phrases were the words with largest deviation for every test sample.

3.2.2 HotFlip

Like one pixel attack [28], a similar method named HotFlip was proposed by Ebrahimi et al. [53]. HotFlip was a white-box attack in text and it relied on an atomic flip operation to swap one token for another based on gradient computation. The authors represented samples as one-hot vectors in input space and a flip operation could be represented by:

\[ \vec{v}_{ijb} = (\vec{0}, \ldots, (0, 0, \ldots, 0, -1, 0, 1), \ldots, 0); (0, \ldots) \quad (12) \]

The eq. (12) means that the \( j \)-th character of \( i \)-th word in a sample was changed from a to b, which were both characters respectively at \( a \)-th and \( b \)-th places in the alphabet. The change from directional derivative along this vector was calculated to find the biggest increase in loss \( J(x, y) \) as follows:

\[ \max \nabla_x J(x, y)^T \cdot \vec{v}_{ijb} = \max_{ijb} \frac{\partial J(b)}{\partial x_{ij}} - \frac{\partial J(a)}{\partial x_{ij}} \quad (13) \]

where \( x_{ij}^{(a)} = 1 \). HotFlip could also be used on character-level insertion, deletion and word-level modification. Although HotFlip performed well on character-level models, only few successful adversarial examples could be generated with one or two flips under the strict constraints.

3.2.3 Optimization-based Method

Considering the limitation of gradient-based methods [23], [42], [53, 55] in black-box case, Alzantot et al. [54] proposed a population-based optimization via genetic algorithm [55, 56] to generated semantically similar adversarial examples. They randomly selected words in the input and computed their nearest neighbors by Euclidean Distance in GloVe embedding space [57]. These nearest neighbors which did not fit within the surrounding were filtered based on language model [58] scores and only high-ranking words with the highest scores were kept. The substitute which would maximize probability of the target label was picked from remaining words. At the same time, aforementioned operations were conducted several times to get a generation. If predicted label of modified samples in a generation were not the target label, the next generation was generated by randomly choosing two samples as parents each time and the same process was repeated on it. This optimization procedure was done to find successful attack by genetic algorithm. In this method, random selection words in the sequence to substitute were full of uncertainty and they might be meaningless for the target label when changed.

3.2.4 Summary of adversarial attacks for classification

These attacks above for classification are either popular or representative ones in recent studies. Some main attributes
of them are summarized in table 1 and instances in these literatures are in appendix A.

3.3 Adversarial examples on other tasks

We have reviewed adversarial attacks for classification task in the previous subsections. But what other kinds of tasks or applications can be attacked by adversarial examples? How are they generated in these cases and whether the crafted examples can be applied in another way except for attack? These questions naturally arise and the answers will be described below.

3.3.1 Attack on Reading Comprehension Systems

In order to know whether reading comprehension systems could really understand language, Jia et al. [62] inserted adversarial perturbations into paragraphs to test the systems without changing the true answers or misleading humans. They extracted nouns and adjectives in the question and replaced them with antonyms. Meanwhile named entities and numbers were changed by the nearest word in GloVe embedding space [57]. The modified question was transformed into declarative sentence as the adversarial perturbation which was concatenated to the end of the original paragraph. This process was call ADDSENT by the authors.

Another process ADDANY was also used to randomly choose any sequence of some words to craft. Compared with ADDSENT, ADDANY did not consider grammaticality and it needed query the model several times. Certainly, both two kinds of generated adversarial examples could fool reading comprehension systems well that gave out incorrect answers. Mainly because they tried to draw the models attention on the generated sequences. Mudrakarta et al. [63] also studied adversarial examples on answering question system and part of their work could strengthen attacks proposed by Jia et al. [62].

3.3.2 Attack on Natural Language Inference (NLI) Models

Besides reading comprehension systems [62], Minervini et al. [64] cast the generation of adversarial examples which violated the given constraints of First-Order Logic (FOL) in NLI as an optimization problem. They maximized the proposed inconsistency loss to search for substitution sets S by using a language model as follows:

\[
\max_{S} J_{1}(S) = \max \{ p(S; body) - p(S; head) \}_{+},
\]

subject to \( \log_{s.t.} p_{L}(S) \leq \tau \)

where \([x]_{+} = \max(0, x)\) and \(\tau\) was a threshold on the perplexity of generated sequences. \(S = X_{1} \rightarrow s_{1}, \ldots, X_{n} \rightarrow s_{n}\) denoted a mapping from \(X_{1}, \ldots, X_{n}\) which was the set of universally quantified variables in a rule to sequences in \(S\). \(p(S; body)\) and \(p(S; head)\) denoted the probability of the given rule, after replacing \(X_{i}\) with the corresponding sentence \(s_{i}\). The generated sequences which were the adversarial examples helped the authors find weaknesses of NLI systems when faced with linguistic phenomena, i.e. negation and antonymy.

3.3.3 Attack on Neural Machine Translation (NMT)

NMT was another kind of system attacked by adversaries and Belinkov et al. [65] made this attempt. They devised black-box methods depending on natural and synthetic language errors to generate adversarial examples. The naturally occurring errors included typos, misspelling words or others and synthetic noise was modified by random or keyboard typo types. These experiments were done on three different NMT systems [66], [67] and results showed that these examples could also effectively fool the target systems.

The same work was also done by Ebrahimi et al. [68] to conduct an adversarial attack on character-level NMT by employing differentiable string-edit operations. The method of generating adversarial examples was same in their previous work [63]. Compared with Belinkov et al. [65], the authors demonstrated that black-box adversarial examples were much weaker than black-box ones in most cases.

3.3.4 Attack with Syntactically Controlled Paraphrase Networks (SCPNS)

Iyyer et al. [69] crafted adversarial examples by the use of SCPNS they proposed. They designed this model for generating syntactically adversarial examples without decreasing the quality of the input semantics. The general process mainly relied on the encoder-decoder architecture of SCPNS. Given a sequence and a corresponding target syntax structure, the authors encoded them by a bidirectional LSTM model and decoded by LSTM model augmented with soft attention over encoded states [70] and the copy mechanism [71]. They then modified the inputs to the decoder, aiming at incorporating the target syntax structure to generate adversarial examples. The syntactically adversarial sentences not only could fool pre-trained models, but also improved the robustness of them to syntactic variation. The authors also used crowdsourced experiment to demonstrate the validity of the generated.

TABLE 1: Summary of attack methods for classification

| Method              | White/Black box | Target/Non-target | Model                  | Metric                |
|---------------------|-----------------|-------------------|------------------------|-----------------------|
| DeepWordBug [24]    | Black box       | Non-target        | LSTM, char-CNN [59]    | Edit Distance        |
| Papernot et al. [42]| White box       | Non-target        | LSTM [10]              |                       |
| Samanta et al. [53] | White box       | Non-target        | CNN                    |                       |
| iAdv-Text [46]      | White box       | Non-target        | LSTM                   | Consine Similarity    |
| Gong et al. [48]    | White box       | Target            | CNN                    | Word Mover Distance(WMD) |
| TextBügger [39]     | Both two        | Non-target        | CNN [60], char-CNN [59]| All metrics except WMD |
| Text-fool [50]      | Both two        | Target            | char-CNN [59]          |                       |
| HotFlip [53]        | White box       | Target            | char-CNN-LSTM [51], CNN [60]| Consine Similarity |
| Alzantot et al. [54]| Black box       | Target            | LSTM [15], RNN [12]   | Euclidean Distance    |

10. https://iamtrask.github.io/2015/11/15/anyone-can-code-lstm/
11. https://github.com/keras-team/keras/blob/master/examples/imdb
12. https://github.com/Smerity/keras/blob/master/sni_mnn.py
3.3.5 Adversarial Examples to Measure Robustness of the Model

Apart from attacks, adversarial examples were used as a way to measure robustness of DNN models. Blohm et al. [73] generated adversarial examples to find out the limitations of a machine reading comprehension model they designed. The categories of adversarial examples included word-level and sentence-level attack in different scenarios [74]. By comparing with human performance, experiment results showed that some other attributions, e.g. answer by elimination via ranking plausibility [74], should be added into this model to improve its performance.

4 Defenses against Adversarial Attacks in Text

The constant arms race between adversarial attacks and defenses invalidates conventional wisdom quickly [25]. In fact, defense is more difficult than attack and few works have been done on this aspect. There are two reasons for this situation. One is that a good theoretical model do not exist for complicated optimization problems like adversarial examples. The other is that tremendous amount of possible inputs may produce the target output with a very high possibility. Hence, a truly adaptive defense method is difficult. In this section, we describe some relatively effective methods of defenses against adversarial attacks in text.

4.1 Defenses by processing training or input data

Adversarial examples are also a kind of data with a special purpose. The first thing to think about is whether data processing or detecting is useful against adversarial attacks. Researchers have done various attempts such as adversarial training and spelling check in text.

4.1.1 Adversarial Training

Adversarial training [8] was a direct approach to defend adversarial images in some studies [8, 25]. They mixed the adversarial examples with corresponding original examples as training dataset to train the model. Adversarial examples could be detected to a certain degree in this way, but adversarial training method was not always work. In text, there were some effects against the attacks after adversarial training [24, 49, 53]. However, it failed in the work of [54], mainly because the different ways of generating adversarial examples. The modifications of the former were insertion, substitution, deletion and replacement, while the later took use of genetic algorithm to search for adversarial examples.

Overfitting may be another reason why adversarial training method is not always useful and may be only effective on its corresponding attack. This has been confirmed by Tram’er et al. [76] in image domain, but it remains to be demonstrated in text.

4.1.2 Spelling Check

Another strategy of defense against adversarial attacks is to detect whether input data is modified or not. Researchers think that there exists some different features between adversarial example and its clean example. For this view, a series of work [72, 73, 78, 80, 83] has been conducted to detect adversarial examples and performs relatively well in image. In text, the ways of modification strategy in some methods may produce misspelling words in generated adversarial examples. This is a distinct different feature which can be utilized. It naturally came up with an idea to detect adversarial examples by checking out the misspelling words. Gao et al. [24] used an autocorrector which was the Python autocorrect 0.3.0 package before the input. And Li et al. [49] took advantage of a context-aware spelling check service [4] to do the same work. But experiment results showed that this approach was effective on character-level modifications and partly useful on word-level operations. Meanwhile, the availability of different modifications was also different no matter on character-level or word-level methods.

4.2 Re-defining function to improve robustness

Except for adversarial training and spelling checking, improving robustness of the model is another way to resist adversarial examples. With the purpose of improving the ranking robustness to small perturbations of documents in the adversarial Web retrieval setting, Goren et al. [82] formally analyzed, defined and quantified the notions of robustness of linear learning-to-rank-based relevance ranking function. They adapted the notions of classification robustness [7, 83] to ranking function and defined related concepts of pointwise robustness, pairwise robustness and a variance conjecture. To quantify the robustness of ranking functions, Kendall’s-τ distance [84] and top change were used as normalized measures. Finally, the empirical findings supported the validity of the authors’ analyses in two families of ranking functions [85, 86].

5 Testing and Verification as the Important Defenses against Adversarial Attacks

The current security situation in DNNs seems to fall into a loop that new adversarial attacks are identified and then followed by new countermeasures which will be subsequently broken [57]. Hence, the formal guarantees on DNNs behavior are badly needed. But it is a hard work and nobody can ensure that their methods or models are perfect. Recently, what we could do is to make the threat of adversarial attacks as little as possible. The technology of testing and verification helps us deal with the problems from another point of view. By the means of it, people can know well about the safety and reliability of systems based on DNNs and determine whether to take measures to address security issues or anything else.

In this section, we introduce recent testing and verification methods for enhancing robustness of DNNs against adversarial attacks. Even though these methods reviewed below have not applied in text, we hope someone interested in this aspect can be inspired and comes up with a good defense method used in text or all areas.
5.1 Testing methods against adversarial examples

As increasingly used of DNNs in security-critical domains, it is very significant to have a high degree of trust in the models' accuracy, especially in the presence of adversarial examples. And the confidence to the correct behavior of the model is derived from the rigorous testing in a variety of possible scenarios. More importantly, testing can be helpful for understanding the internal behaviors of the network, contributing to the implementation of defense methods. This applies the traditional testing methodology used in DNNs.

5.1.1 Coverage-driven Testing

Pei et al. [88] designed a white-box framework DeepXplore to test real-world DNNs with the metric of neuron coverage and leveraged differential testing to catch the differences of corresponding output between multiple DNNs. In this way, DeepXplore could trigger the majority logic of the model to find out incorrect behaviors without manual efforts. It performed well in the advanced deep learning systems and found thousands of corner cases which would make the systems crash. However, the limitation of DeepXplore was that if all the DNNs made incorrect judgement, it was hard to know where was wrong and how to solve it.

Different from single neuron coverage [88], Ma et al. [89] proposed a multi-granularity testing coverage criteria to measure accuracy and detect erroneous behaviors. They took advantage of four methods [8], [27], [29], [33] to generate adversarial test data to explore the new internal states of the model. The increasing coverage showed that the larger the coverage was, the more possible the defects were to be checked out. Similar work was done by Budnik et al. [90] to explore the output space of the model under test via an adversarial case generation approach.

In order to solve the limitation of neuron coverage, Kim et al. [91] proposed a Surprise Adequacy for Deep Learning Systems(SADL) to test DNNs and developed Surprise Coverage(SC) to measure the coverage of the range of Surprise Adequacy(SA) values, which measured the different behaviors between inputs and training data. Experimental results showed that the SA values could be a metric to judge whether an input was adversarial example or not. In other hand, it could also improve the accuracy of DNNs against adversarial examples by retraining.

5.1.2 Feature-guided Testing

There also exists other kinds of testing method against adversarial examples. Wicker et al. [92] presented a feature-guided approach to test the resilience of DNNs in black-box scenario against adversarial examples. They treated the process of generating adversarial cases as a two-player turn-based stochastic game with the asymptotic optimal strategy based on Monte Carlo tree search (MCTS) algorithm. In this strategy, there was an idea of reward for accumulating adversarial examples found over the process of game play and evaluated the robustness against adversarial examples by the use of it.

5.1.3 Concolic Testing

Besides the feature-guided testing [92], Sun et al. [93] presented DeepConcolic to evaluate the robustness of well-known DNNs, which was the first attempt to apply traditional concolic testing method for these networks. DeepConcolic iteratively used concrete execution and symbolic analysis to generate test suit to reach a high coverage and discovered adversarial examples by a robustness oracle. The authors also compared with other testing methods [88], [89], [94], [95]. In terms of input data, DeepConcolic could start with a single input to achieve a better coverage or used coverage requirements as inputs. In terms of performance, DeepConcolic could achieve higher coverage than DeepXplore, but run slower than it.

5.2 Verification methods against adversarial examples

Researchers think that testing is insufficient to guarantee the security of DNNs, especially with unusual inputs like adversarial examples. As Edsger W. Dijkstra once said, testing shows the presence, not the absence of bugs. Hence, verification techniques on DNNs are needed to study more effective defense methods in adversarial settings.

Pulina et al. [96] might be the first to develop a small verification system for a neural network. Since then, related work appears one after another. But verification of machine learning models robustness to adversarial examples is still in its infancy [97]. There is only a few researches on related aspects. We will introduce these works in the following part.

5.2.1 Approaches based on Satisfiability Modulo Theory

There are several researches to check security properties against adversarial attacks by diverse kinds of Satisfiability Modulo Theory (SMT) solvers. Katz et al. [99] presented a novel system named Reluplex to verify DNNs by splitting the problem into the LP problems with Rectified Linear Unit (ReLU) activation functions based on SMT solver. Reluplex could be used to find adversarial inputs with the local adversarial robustness feature on the ACAS Xu networks, but it failed on large networks on the global variant.

Huang et al. [101] proposed a new verification framework which was also based on SMT to verify neural network structures. It relied on discretizing search space and analyzing output of each layer to search for adversarial perturbations, but the authors found that SMT theory could only suitable for small networks in practice. On the other hand, this framework was limited by many assumptions and some of functions in it were unclear.

5.2.2 Approaches based on Mixed Integer Linear Programming

For ReLU networks, a part of researches regarded the verification as a Mixed Integer Linear Programming (MILP) problem such as Tjeng et al. [102]. They evaluated robustness to adversarial examples from two aspects of minimum adversarial distortion [103] and adversarial test accuracy [104]. Their work was faster than Reluplex with a high adversarial test accuracy, but the same limitation was that it remained a problem to scale it to large networks.

Different from other works, Narodytska et al. [105] verify the secure properties on the binarized neural networks (BNNs) [106]. They were the first to utilize exact Boolean encoding on a network to study its robustness and
equivalence. The inputs would be judged whether they were adversarial examples or not by two encoding structures Gen and Ver. It could easily find adversarial examples for up to 95 percent of considered images on the MNIST dataset and also worked on the middle-sized BNNs rather than large networks.

5.2.3 Other methods with ReLU activation functions

There is a different point of view that the difficulty in proving properties about DNNs is caused by the presence of activation functions [99]. Some researchers pays more attention to them for exploring better verification methods.

Gehr et al. [107] introduced abstract transformers which could get the outputs of layers in convolutional neural network with ReLU, including fully connected layer. The authors evaluated this approach on verifying robustness of DNNs such as pre-trained defense network [108]. Results showed that FGSM attack could be effectively prevented. They also did some comparisons with Reluplex on both small and large networks. The stare-of-the-art ReluV al performed worse than it in verification of properties and time consumption.

Unlike existing solver-based methods (e.g. SMT), Wang et al. [109] presented ReLuVal which leveraged interval arithmetic [110] to guarantee the correct operations of DNNs in the presence of adversarial examples. They repeatedly partitioned input intervals to find out whether the corresponding output intervals violated security property or not. By contrast, this method was more effective than Reluplex and performed well on finding adversarial inputs.

Weng et al. [111] designed two kinds of algorithm to evaluate lower bounds of minimum adversarial distortion via linear approximations and bounding the local Lipschitz constant. Their methods could be applied into defended networks especially for adversarial training to evaluate the effectiveness of them.

6 Discussion of Challenges and Future Direction

In the previous sections, a detailed description of adversarial examples on attack and defense was given to enable readers to have a faster and better understanding of this respect. Next, we present more general observations and discuss challenges on this direction based on the aforementioned contents.

Judgement on the performance of attack methods: Generally, authors mainly evaluate their attacks on target models by accuracy rate or error rate. The lower the accuracy rate is, the more effective the adversarial examples are. And the use of error rate is the opposite. Certainly, some researchers prefer to utilize the difference in accuracy before and after attacks, because it can show the effect of attacks more intuitively. And these criterions can also used in defending of adversarial examples.

Reasons by using misspelled words in some methods: The motivation by using misspelled words is similar to that in image, which aims at fooling target models with indiscernible perturbations. Some methods tend to conduct character-level modification operations which highly result in misspelled words. And humans are extremely robust against that case in written language [112].

Transferability in black-box scenario: When the adversaries have no access including probing to the target models, they train a substitute model and utilize the transferability of adversarial examples. Szegedy et al. [7] first found that adversarial examples generated from a neural network could also make another model misbehave by different datasets. This reflects the transferability of the adversarial example. As a result, adversarial examples generated in the substitute model are used to attack the target models while models and datasets are all inaccessible. Apart from that, constructing adversarial examples with high transferability is a prerequisite to evaluate the effectiveness of black-box attacks and a key metric to evaluate generalized attacks [113].

The lack of a universal approach to generate adversarial examples: Because the application of adversarial examples in text rose as a frontier in recent years, the methods of adversarial attacks were relatively few, let alone defenses. The another reason why this kind of method do not exist is the language problem. Almost all recent methods use English dataset and the generated adversarial examples may be useless to the systems with Chinese or other language dataset. Thus, there is not a universal approach to generate adversarial examples. But in our observations, many methods mainly follow a two-step process to generate adversarial examples. The first step is to find important words which have significant impact on classification results and then homologous modifications are used to get adversarial examples.

Difficulties on adversarial attacks and defenses: There are many reasons for this question and one of the main reasons is that there is not a straightforward way to evaluate proposed works no matter attack or defense. Namely, the convincing benchmarks do not exist in recent works. One good performed attack method in a scenario may failed in another or new defense will soon be defeated in the way beyond defenders’ anticipation. Even though some works are provably sound, but rigorous theoretical supports are still needed to deal with the problem of adversarial examples.

Appropriate future directions on adversarial attacks and defenses: As an attacker, designing universal perturbations to catch better adversarial examples can be taken into consideration like it works in image [5]. A universal adversarial perturbation on any text is able to make a model misbehave with high probability. Moreover, more wonderful universal perturbations can fool multi-models or any model on any text. On the other hand, the work of enhancing the transferability of adversarial examples is meaningful in more practical back-box attacks. On the contrary, defenders prefer to completely revamp this vulnerability in DNNs, but it is no less difficult than redesigning a network and is also a long and arduous task with the common efforts of many people. At the moment defender can draw on methods from image area to text for improving the robustness of DNNs, e.g. adversarial training [108], adding extra layer [114], optimizing cross-entropy function [115], [116] or weakening the transferability of adversarial examples.
7 Conclusion

This article presents a survey about adversarial attacks and defenses on DNNs in text. Even though DNNs have the high performance on a wide variety of NLP, they are inherently vulnerable to adversarial examples, which lead to a high degree concern about it. This article integrates almost existing adversarial attacks and some defenses focusing on recent works in the literature. From these works, we can see that the threat of adversarial attacks is real and defense methods are few. Most existing works have their own limitations such as application scene, constraint condition and problems with the method itself. More attention should be paid on the problem of adversarial example which remains an open issue for designing considerably robust models against adversarial attacks.

Acknowledgment

This work was partly supported by NSFC under No. 61876134, the National Key R&D Program of China under No. 2016YFB0801100, NSFC under U1536204 and U183610015.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in NIPS’12 Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, 2012, pp. 1097–1105.

[2] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in NIPS’15 Proceedings of the 28th International Conference on Neural Information Processing Systems, vol. 1, 2015, pp. 91–99.

[3] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. rahan Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97, 2012.

[4] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent Neural Networks,” in NIPS’13 Proceedings of the 25th International Conference on Neural Information Processing Systems, vol. 1, 2013, pp. 1097–1105.

[5] K. Chen, Y. Wang, L. Cui, and Y. Han, “An efficient approach to train very deep neural network,” Proceedings of the 29th International Conference on Neural Information Processing Systems, vol. 1, 2015, pp. 91–99.

[6] H. Xue, C. Dong, D. Zhu, A. Kotov, A. I. Carbone, and S. NaaKing, “Text classification with topic-based word embedding and convolutional neural networks,” in BCN ’16 Proceedings of the 9th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics, 2016, pp. 88–97.

[7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” in Proceedings of the International Conference on Learning Representations, 2014.

[8] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” in Proceedings of the International Conference on Learning Representations, 2015.

[9] N. Akhtar and A. Mian, “Threat of adversarial attacks on deep learning in computer vision: A survey,” IEEE Access, vol. 6, pp. 14410 – 14430, 2018.

[10] A. P. Norton and Y. Qi, “Adversarial-playground: A visualization suite showing how adversarial examples fool deep learning,” in IEEE Symposium on Visualization for Cyber Security, 2017, pp. 206–208.

[11] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, “Practical black-box attacks against machine learning,” in Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, 2017, p. 506519.

[12] S. Shen, R. Furuta, T. Yamashita, and K. Aizawa, “Fooling neural networks in face attractiveness evaluation: Adversarial examples with high attractiveness score but low subjective score,” in IEEE Third International Conference on Multimedia Big Data, IEE, 2017.

[13] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, “Robust physical-world attacks on deep learning models,” in the 31th IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[14] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, “Adversarial examples for semantic segmentation and object detection,” in IEEE International Conference on Computer Vision, 2017, pp. 1397–1398.

[15] R. Taori, A. Kasemittay, B. Chu, and N. Vemuri, “Targeted adversarial examples for black box audio systems,” 2018, arXiv preprint arXiv: 1805.07820.

[16] N. Carlini and D. Wagner, “Audio adversarial examples: Targeted attacks on speech-to-text,” in IEEE Security and Privacy Workshops, 2018.

[17] H. Yan, Y. Yin, and J. Zhang, “Robust audio adversarial example for physical attack,” 2018, arXiv preprint arXiv: 1810.11793.

[18] L. Liu, H. Li, and J. Zhang, “Adversarial examples: Attacks on machine learning-based malware visualization detection methods,” 2018, arXiv preprint arXiv:1808.01546.

[19] W. He and Y. Tan, “Generating adversarial malware examples for black-box attacks based on gan,” 2017, arXiv preprint arXiv: 1702.05983.

[20] W. Medhat, A. Hassan, and H. Korashy, “Sentiment analysis algorithms and applications: A survey,” Ain Shams Engineering Journal, vol. 5, no. 4, pp. 1093–1113, 2014.

[21] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, “Abusive language detection in online user content,” in WWW ’16 Proceedings of the 25th International Conference on World Wide Web, 2016, pp. 145–153.

[22] S. Rayana and L. Noghi, “Collective opinion spam detection: Bridging review networks and metadata,” in ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, 2015.

[23] X. Xiao, B. Yang, and Z. Kang, “A gradient tree boosting based approach to rumor detecting on sina weibo,” 2018, arXiv preprint arXiv:1806.06326.

[24] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, “Black-box generation of adversarial text sequences to evade deep learning classifiers,” in IEEE Security and Privacy Workshops (SPW). IEEE, 2018.

[25] X. Ling, S. Ji, J. Zou, J. Wang, C. Wu, B. Li, and T. Wang, “Deepesec: A uniform platform for security analysis of deep learning model,” in IEEE Symposium on Security and Privacy (SP), 2019, pp. 381–398.

[26] M. Naseer, S. H. Khan, S. Rahman, and F. Porikli, “Distorting neural representations to generate highly transferable adversarial examples,” 2018, arXiv preprint arXiv: 1811.09020.

[27] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” in IEEE European Symposium on Security and Privacy. IEEE, 2016.

[28] J. Su, D. V. Vargas, and S. Kouichi, “One pixel attack for fooling deep neural networks,” 2017, arXiv preprint arXiv:1710.08864.

[29] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in IEEE Symposium on Security and Privacy (SP). IEEE, 2017.

[30] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2017.

[31] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, “Deepfool: a simple and accurate method to fool deep neural networks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2574–2582.

[32] M. Cisse, Y. Adi, N. Neverova, and J. Keshet, “Houdini: Breaking high performance image classifiers,” 2017, arXiv preprint arXiv: 1707.05373.

[33] A. Kurakin, I. Goodfellow, and S. Bengio, “Adversarial examples in the physical world,” in Proceedings of the International Conference on Learning Representations, 2017.

[34] S. Sarkar, A. Bansal, U. Mahbub, and R. Chellappa, “Upset and angi: Breaking high performance image classifiers,” 2017, arXiv preprint arXiv:1707.01159.
[35] S. Baluja and I. Fischer, “Adversarial transformation networks: Learning to generate adversarial examples,” 2017, arXiv preprint arXiv:1703.09387.

[36] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” vol. 13, no. 4, pp. 600–611, 2004.

[37] B. Luo, Y. Liu, L. Wei, and Q. Xu, “Towards imperceptible and robust adversarial example attacks against neural networks,” in Proceedings of Association for the Advancement of Artificial Intelligence, 2018.

[38] M. J. Kusner, Y. Sun, N. I. Kolkin, and K. Q. Weinberger, “From word embeddings to document distances,” in Proceedings of the International Conference on International Conference on Machine Learning, 2015, pp. 957–966.

[39] Y. Rubner, C. Tomasi, and L. J. Guibas, “A metric for distributions with applications to image databases,” in ICCV ’98 Proceedings of the Sixth International Conference on Computer Vision. IEEE, 1998, pp. 59–66.

[40] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals,” vol. 10, no. 8, pp. 707–710, 1966.

[41] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, “A large annotated corpus for learning natural language inference,” in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2015.

[42] N. Papernot, P. McDaniel, A. Swami, and R. Harang, “Crafting adversarial input sequences for recurrent neural networks,” in IEEE Military Communications Conference, 2016, p. 4954.

[43] P. J. Werbos, “Generalization of backpropagation with application to a recurrent gas market model,” Neural Networks, vol. 1, no. 4, pp. 339–356, 1988.

[44] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[45] S. Samanta and S. Mehta, “Towards crafting text adversarial samples,” 2017, arXiv preprint arXiv:1707.02812.

[46] M. Sato, J. Suzuki, H. Shindo, and Y. Matsumoto, “Interpretable adversarial perturbation in input embedding space for text,” in International Joint Conference on Artificial Intelligence (IJCAI). 2018.

[47] T. Miyato, A. M. Dai, and I. Goodfellow, “Adversarial training methods for semi-supervised text classification,” in Proceedings of the International Conference on Learning Representations, 2017.

[48] Z. Gong, W. Wang, B. Li, D. Song, and W.-S. Ku, “Adversarial texts with gradient methods,” 2018, arXiv preprint arXiv:1801.07173.

[49] J. Liu, S. Ji, T. Du, and T. Wang, “Textbugger: Generating adversarial text against real-world applications,” in the Network and Distributed System Security Symposium, 2019.

[50] B. Liang, H. Li, M. Su, P. Bian, X. Li, and W. Shi, “Deep text classification can be fooled,” in Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, 2018, p. 4208–4215.

[51] M. N. Alshalah, V. Raskin, M. Crogan, C. Hempellman, F. Kerschbaum, D. Mohamed, and S. Naik, “Natural language watermarking: Design, analysis, and a proof-of-concept implementation,” in Information Hiding, p. 185200, 2001.

[52] M. Sutton, A. Greene, and P. Amini, Fuzzing: Brute Force Vulnerability Discovery. Addison-Wesley Professional, 2007.

[53] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, “Hotflip: White-box adversarial examples for text classification,” in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. ACL, 2018, pp. 31–36.

[54] M. Alzantot, Y. Sharma, A. Elgohary, B.-J. Ho, M. Srivastava, and K.-W. Chang, “Generating natural language adversarial examples,” in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018.

[55] E. J. Anderson and M. C. Ferris, “Genetic algorithms for combinatorial optimization: the assemble line balancing problem,” ORSA Journal on Computing, vol. 6, no. 2, p. 161173, 1994.

[56] H. Mihlenbein, “Parallel genetic algorithms, population genetics and combinatorial optimization,” in Proceedings of the third international conference on Genetic algorithms, 1989, pp. 416–421.

[57] J. Pennington, R. Socher, and C. Manning, “Glove: Global vectors for word representation,” in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, p. 15321543.

[58] C. Chelsea, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, “One billion word benchmark for measuring progress in statistical language modeling,” Computer Science, 2013.

[59] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” in Advances in neural information processing systems, 2015, p. 649657.

[60] Y. Kim, “Convolutional neural networks for sentence classification,” in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014.

[61] Y. Kim, Y. Jernite, D. Sontag, and A. de R Rush, “Character-aware neural language models,” in Proceedings of AAAI, 2016.

[62] R. Jia and P. Liang, “Adversarial examples for evaluating reading comprehension systems,” in Proceedings of the 2017 conference on empirical methods in natural language processing (EMNLP), 2017, p. 20212031.

[63] P. K. Mudrakarta, A. Tal, M. Sundararajan, and K. Dhamdhere, “Did the model understand the question?” in the 56th Annual Meeting of the Association for Computational Linguistics, 2018.

[64] P. Minervini and S. Riedel, “Adversarially regularising neural nli models to integrate logical background knowledge,” in the SIGNLL Conference on Computational Natural Language Learning. 2018.

[65] Y. Belinkov and Y. Bisk, “Synthetic and natural noise both break neural machine translation,” in Proceedings of the International Conference on Computational Linguistics, 2018.

[66] J. Lee, K. Cho, and T. Hofmann, “Fully character-level neural machine translation without explicit segmentation,” in Transactions of the Association for Computational Linguistics (TACL), 2017.

[67] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Htucher, M. Junczys-Dowmunt, S. Laubli, A. V. M. Barone, J. Moky, and N. Ndejde, “Nematus: a toolkit for neural machine translation,” in Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, 2017, p. 6508.

[68] J. Ebrahimi, D. Lowd, and D. Dou, “On adversarial examples for character-level neural machine translation,” in Proceedings of the 27th International Conference on Computational Linguistics, 2018.

[69] M. Iyyer, J. Wieting, K. Gimpel, and L. Zettlemoyer, “Adversarial example generation with syntactically controlled paraphrase networks,” in Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics, 2018.

[70] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in Proceedings of the International Conference on Learning Representations, 2014.

[71] A. See, P. J. Liu, and C. D. Manning, “Get to the point: Summarization with pointer-generator networks,” in Proceedings of the Association for Computational Linguistics, 2017.

[72] M. Blohm, G. Jagfeld, E. Sood, X. Yu, and N. T. Vu, “Comparing attention-based convolutional and recurrent neural networks: Success and limitations in machine reading comprehension,” in Proceedings of the SIGNLL Conference on Computational Natural Language Learning, 2018.

[73] X. Yuan, P. He, Q. Zhu, and X. Li, “Adversarial examples: Attacks and defenses for deep learning,” pp. 1–20, 2019.

[74] J. E. Hummel and K. J. Holyoak, “Relational reasoning in a flexible learning system,” in Proceedings of the 7th Annual Meeting of the Association for Computational Linguistics.

[75] C. K. Mumtadj, T. Brox, and J. H. Metzen, “Defending against universal perturbations with shared adversarial training,” 2018, arXiv preprint arXiv:1812.03705.

[76] F. Tram’er, A. Kurakin, N. Papernot, D. Boneh, and P. McDaniel, “Ensemble adversarial training: Attacks and defenses,” 2018, arXiv preprint arXiv:1703.00410.
### 8 Appendix A

Part 1: Instances of Adversarial Examples

**TABLE 2: Instances of some methods in section 3**

| Method          | Instance of adversarial example                                      | Modification                                      |
|-----------------|---------------------------------------------------------------------|---------------------------------------------------|
| DeepWordBug     | original: This film has a special place in my heart. **positive**    | modified: This film has a special **plcae** in my **herat. negative** |
| Papernot et al. | original: I wouldn’t rent this one even on dollar rental night. **negative** | modified: **Excellent** wouldn’t rent this one even on dollar rental night. **positive** |
| Samanta et al.  | original: A sprawling, overambitious, plotless comedy that has no dramatic center. It was probably intended to have an epic vision and a surrealistic flair (at least in some episodes), but the separate stories are never elevated into a **greatly** whole, and the laughs are **little** and far between amusing ending though. **positive** |
| iAdv-Text       | original: The essence of this film falls on judgments by police officers who fortunately ethical and moral men act on situations within situations in a city with a super abundance of violence and killing Good compound interacting story lines and above average charactarizations. **positive** modified: The essence from **THIS** film falls on judgments by police officers who fortunately ethical and moral men act on situations within situations in a city with a super abundance of violence and killing Good compound interacting story lines and above average charactarizations. **negative** |
| Gong et al.      | original: FED EXPECTED TO ADD RESERVES The Federal Reserve will enter the government securities market to supply reserves via either a large round of customer repurchase agreements or by overnight or possibly four-day system repurchases, economists said. [ .. ] tax payments swell Treasury balances at the Fed . Fed funds hovered at 6-3 / 4 pct after averaging 6 . 80 pct yesterday. **Label 2** modified: FED EXPECTED TO ADD RESERVES The Federal Reserve will enter the government securities market **able** supply reserves via either a large round of customer repurchase agreements or **directed** overnight or possibly four-day system repurchases books economists said . [ .. ] tax payments surf Treasury balances at the **future** well Fed funds hovered at 6-3 **Category** 4 **Q2** after averaged 6 . 80 pct yesterday. **Label 3** |
| TextBugger      | original: reason why requesting i want to report something so can ips report stuff, or can only registered users can? if only registered users can, then i ’ll request an account and it ’s just not fair that i cannot edit because of this anon block shit c’mon, fucking hell. **Toxic** modified: reason why requesting i want to report something so can ips report stuff, or can only registered users can? if only registered users can, then i ’ll request an account and it ’s just not fair that i cannot edit because of this anon block shit c’mon, fucking hell. **Non-toxic** |
| Text-fool       | original: The Old Harbor Reservation Parkways are three historic roads in the Old Harbor area of Boston. They are part of the Boston parkway system designed by Frederick Law Olmsted. They include all of William J. Day Boulevard running from Castle Island to Kosciuszko Circle along Pleasure Bay and the Old Harbor shore. The part of Columbia Road from its northeastern end at Farragut Road west to Pacuska Circle (formerly called Preble Circle). 87.3%. **Building** modified: The Old Harbor Reservation Parkways are three **historic** roads in the Old Harbor area of Boston. Some exhibitions of Navy aircrafts were held here. They are part of the Boston parkway system designed by Frederick Law Olmsted. They include all of William J. Day Boulevard running from Castle Island to Kosciuszko Circle along Pleasure Bay and the Old Harbor shore. The part of Columbia Road from its northeastern end at Farragut Road west to Pacuska Circle formerly called Preble Circle. 95.7%. **Means of Transportation** |
| HotFlip         | original: South Africans historic Soweto township marks its 100th birthday on Tuesday in a mood of optimism. 57%. **World** modified: South Africans historic Soweto township marks its 100th birthday on Tuesday in a mood of optimism. 95%. **Sci/Tech** |
| Alzantot et al. | premise: A runner wearing purple strives for the finish line. **86%Entailment** original: A runner wants to head for the finish line. **43%Contradiction** | modified: A **racer** wants to head for the finish line. **Non-toxic** |