applied sciences

MDPI

Article

CRank: Reusable Word Importance Ranking for Text Adversarial Attack

Xinyi Chen † and **Bo Liu** *

Computer College, National University of Defense Technology, Changsha 410000, China; xinyi_chen@nudt.edu.cn

* Correspondence: kyle.liu@nudt.edu.cn

† Current address: Deya Road 109, Kaifu District, Changsha 410000, China.

Abstract: Deep learning models have been widely used in natural language processing tasks, yet researchers have recently proposed several methods to fool the state-of-the-art neural network models. Among these methods, word importance ranking is an essential part that generates text adversarial examples, but suffers from low efficiency for practical attacks. To address this issue, we aim to improve the efficiency of word importance ranking, making steps towards realistic text adversarial attacks. In this paper, we propose CRank, a black box method utilized by our innovated masking and ranking strategy. CRank improves efficiency by 75% at the 'cost' of only a 1% drop of the success rate when compared to the classic method. Moreover, we explore a new greedy search strategy and Unicode perturbation methods.

Keywords: deep neural networks; natural language processing; adversarial examples

Citation: Chen, X.; Liu, B. CRank: Reusable Word Importance Ranking for Text Adversarial Attack. *Appl. Sci.* **2021**, *11*, 9570. https://doi.org/10.3390/app11209570

Academic Editor: David Megías

Received: 30 August 2021
Accepted: 13 October 2021
Published: 14 October 2021

1. Introduction

Despite the impressive success of deep neural networks (DNNs), researchers have found their vulnerability. Exploiting such vulnerabilities, also known as adversarial attacks, aims to generate adversarial examples by adding imperceptible perturbations to normal samples. Such generated adversarial examples bring no misunderstandings to humans, while fooling the neural network models to make wrong predictions. Representative works are first proposed in computer vision (CV) [1–3]. In the context of NLP, the adversarial attack is more challenging due to the discrete nature of the text, as editing a single word might change the entire meaning of a text while changing limited pixels in an image is not obvious for humans.

Recently, several studies have demonstrated different text adversarial attacks against deep neural networks in a variety of natural language processing (NLP) tasks [4–7]. Among these studies, there is a trend involving black box attacks that are agnostic to the target model, except the input and output. Such attacks query the target model with continuously improving examples, to find a successful adversarial example. They firstly rank words to find those that have a big impact on the target model. These ranking methods, referred to as word importance ranking (WIR) [8], generally rank a word by deleting it or replacing it with a certain string, then query the target model with the modified sentence for its score. However, present WIR methods encounter a problem where they need hundreds of queries to generate one successful adversarial example, which makes them unpractical in attacking real-world applications. Such inefficiency brings us two questions: can we find an alternative solution that greatly improves efficiency? How many side effects does the alternative solution have if found?

With these research questions, we review classic WIR methods in representative works [7–11], and conclude a critical defect: classic methods consume multiple queries for the same word if the word shows up in different sentences. Thus, we are motivated to create a new method that only needs one query for a word.

In this paper, we propose a reusable and efficient black box WIR method, CRank. CRank uses a special strategy that only needs one query to score a word, even when the word exists in many sentences. Such strategy masks every word, except the target word, while classic methods only mask the target word.

Our main contribution is summarized as follows:

- We firstly introduce a three-step workflow, word importance ranking, search, and perturbation for the text adversarial attack. Our workflow is clearer than classic ones that emphasize the two-step attack, search, and perturbation.
- We present CRank and compare it with the classic method with Word-CNN and Word-LSTM on three different datasets. Experimental results reveal that CRank reduces queries by 75% while achieving a similar success rate that is only 1% lower.
- We explore other improvements of the text adversarial attack, including the greedy search strategy and Unicode perturbation methods.

The rest of the paper is organized as follows. The literature review is presented in Section 2 followed by preliminaries used in this research. The proposed approach and experiment are in Sections 4 and 5. Section 6 discusses the limitations and considerations of the approach. Finally, Section 7 draws conclusions and outlines future work.

2. Related Work

Deep learning models have achieved impressive success in many fields, such as healthcare [12], engineering projects [13], cyber security [14], CV [15,16], NLP [17–19], etc. However, these models seem to have inevitable vulnerability and adversarial examples [1,2,20,21], as firstly studied in CV, to fool neural network models while being imperceptible for humans.

In the context of NLP, the initial research [22,23] started with the Stanford Question Answering Dataset (SQuAD) and further works extend to other NLP tasks, including classification [4,7–11,24–27], text entailment [4,8,11], and machine translation [5,6,28]. Some of these works [10,24,29] adapt gradient-based methods from CV that need full access to the target model. An attack with such access is a harsh condition, so researchers explore black box methods that only acquire the input and output of the target model.

Present black box methods rely on queries to the target model and make continuous improvements to generate successful adversarial examples. Gao et al. [7] present effective DeepWordBug with a two-step attack pattern, searching for important words and perturbing them with certain strategies. They rank each word from the original examples by querying the model with the sentence where the word is deleted, then use character-level strategies to perturb those top-ranked words to generate adversarial examples. TextBugger [9] follows such a pattern, but explores a word-level perturbation strategy with the nearest synonyms in GloVe [30]. Later studies [4,8,25,27,31] of synonyms argue about choosing proper synonyms for substitution that do not cause misunderstandings for humans. Although these methods exhibit excellent performance in certain metrics (high success rate with limited perturbations), the efficiency is rarely discussed. Our investigation finds that state-of-the-art methods need hundreds of queries to generate only one successful adversarial example. For example, the BERT-Attack [11] uses over 400 queries for a single attack. Such inefficiency is caused by the classic WIR method that generally ranks a word by replacing it with a certain mask and scores the word by querying the target model with the altered sentence. The method is still used in many state-of-the-art black box attacks, yet different attacks may have different masks. For example, DeepWordBug [7] and TextFooler [8] use an empty mask that is equal to deleting the word, while BERT-Attack [11] and BAE [25] use an unknown word, such as '(unk)' as the mask. However, the classic WIR method encounters an efficiency problem, where it consumes duplicated queries to the same word if the word appears in different sentences.

Despite the work in CV and NLP, there is a growing number of research ib the adversarial attack in cyber security domains, including malware detection [32–34], intrusion detection [35,36], etc. Such facts suggest that the vulnerability of neural network models widely exists. However, the amount of defensive research [37–41] against the adversarial

attack is increasing. In the future, attack and defense methods of adversarial examples will advance together.

3. Preliminaries

This section provides several preliminaries that are used in the following paper, including our research domain, notations, and other necessary knowledge.

3.1. Text Classification

Text classification is a major task in NLP, with many applications, such as sentiment analysis, topic labeling, toxic detection, and so on. Currently, neural network models including convolutional neural networks (CNN), the long short-term memory (LSTM) network, and BERT [42] are widely used in many text classification datasets. Among these datasets, *SST-2* (https://nlp.stanford.edu/sentiment/, accessed on 1 May 2021), *AG News* (http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html, accessed on 1 May 2021), and *IMDB* (http://ai.stanford.edu/~amaas/data/sentiment/, accessed on 1 May 2021) are the most known datasets for various benchmarks. AG News is a sentence-level multi-classification dataset with four news topics: world, sports, business, and science/technology. IMDB and SST-2 are both sentiment binary classification datasets. IMDB is a document-level movie review dataset with long paragraphs and SST-2 is a sentence-level phrase dataset. Three examples of these datasets are demonstrated in Table 1.

Table 1. Dataset Examples.

Dataset	Example	Label
SST-2	The most hopelessly monotonous film of the year, noteworthy only for the gimmick of being filmed as a single unbroken 87-min take.	Negative
AG News	European spacecraft prepares to orbit Moon; Europe's first lunar spacecraft is set to go into orbit around the Moon on Monday. SMART-1 has already reached the gateway to the Moon, the region where its gravity starts to dominate that of the Earth.	Sci/tech
IMDB	The last good Ernest movie, and the best at that. How can you not laugh at least once during this movie? The last line is a classic and showcases Ernest's gangster impressions—his best moment on film. This movie has his best lines, and it is a crowning achievement among the brainless screwball comedies.	Positive

3.2. Threat Model

We study text adversarial examples against text classification under the black box setting, meaning that the attacker is not aware of the model architecture, parameters, or training data, but capable of querying the output of the target model with supplied inputs. The output includes the predictions and their confidence scores. Our method is interactive, which means it needs to repeatedly query the target model with improved inputs to generate satisfying adversarial examples. We perform the non-targeted attack, considering any adversarial example that causes successful misclassification.

3.3. Formulation

We use X to represent the original sentence and Y as its corresponding label. Sentence X is composed of N words $\{W_1, W_2, \ldots, W_N\}$. When we perturb kth word W_k, it becomes W_k' and the new sentence is X'. We use $F : X \Rightarrow Y$ to represent the prediction of the model, and $Conf(X)$ to represent the confidence of X with its original label. For adversarial examples, they should satisfy the following equation:

$$F(X) = Y, and\ F(X') \neq Y \tag{1}$$

Under binary classification tasks, Equation (1) can be presented with confidence scores, as Equation (2) demonstrates.

$$Conf(X) > 0.5, and\ Conf(X') < 0.5 \tag{2}$$

3.4. Unicode

Traditionally, text data are stored in 256 ASCII characters, among which, only 100 characters are printable, including digits, alphabet, and punctuation. Unicode is a more abundant standard for text, which can represent symbols, emojis, different languages, and so on. The most commonly used Unicode standard, UTF-8, uses one to four bytes for a character and theoretically represents 2^{16}(65,536) characters. Nowadays, most websites and social media support Unicode, from which text data are collected for further processing by neural network models. Such facts suggest that it is available to fool text classifiers with Unicode characters. In this paper, we utilize them for the effective adversarial attack.

4. Proposed Approach

As Figure 1 demonstrates, we elaborate our approach with three parts—word importance ranking, search strategies, and perturbation methods. With a given sentence of N words, $X = \{W_1, W_2, \ldots, W_N\}$, word importance ranking aims to sort these words according to their 'importance' to the target model. After we rank word indexes $\{R_1, R_2, \ldots, R_N\}$, we use search strategies to search until a suitable sequence of words to perturb is found. When we find the sequence, we perturb those words and attempt to generate a successful adversarial example X_{adv}.

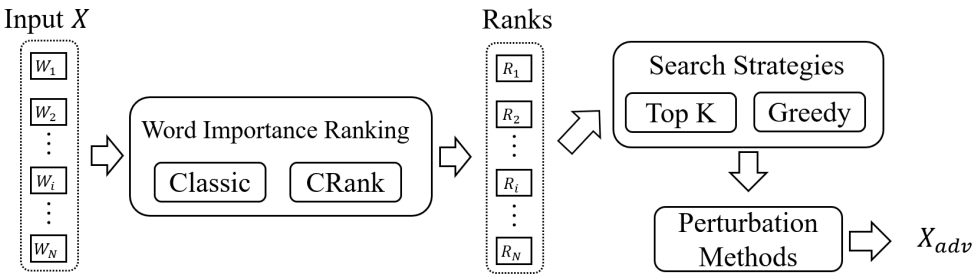

Figure 1. Overview of Our Workflow.

4.1. Word Importance Ranking

In this section, we propose three ranking methods, Classic, CRank, and CRankPlus. Classic is a generic method that is commonly adopted in recent black box researches [7,8,11,25]. CRank is our innovative method, aimed at improving efficiency, and CRankPlus is its improved version with dynamic adjustment.

4.1.1. Classic

The generic word importance ranking method masks each word with W_{mask} to generate example X'_i as Equation (3) demonstrates, then use Equation (4) to calculate its score. We use an unknown word, such as '(mask)' as W_{mask} in our approach, and demonstrate an example of classic word importance ranking in Table 2.

$$X'_i = \{\ldots, W_{i-1}, W_{mask}, W_{i+1}, \ldots\} \tag{3}$$

$$Score(W_i) = Conf(X) - Conf(X'_i) \tag{4}$$

Table 2. Example of classic word importance ranking. The target model is CNN trained with SST-2.

Masked	Sentence	Confidence	Score
(Original)	or doing last year's taxes with your ex-wife.	0.96	
taxes	or doing last year's [mask] with your ex-wife.	0.71	0.25
year's	or doing last [mask] taxes with your ex-wife.	0.84	0.14
ex-wife	or doing last year's taxes with your [mask].	0.88	0.08

4.1.2. CRank

Instead of masking the target word, we rank the word in a 'reverse' way by masking other words in the sentence, as Equation (5) shows, then use Equation (6) to calculate its score. To make CRank reusable, we set the masks with a fixed length of 6. (This result is supported in our experiment in Section 5.3), as our investigation finds that longer sequences of masks will not affect the score, while shorter ones do. As Table 3 demonstrates, we propose four types of CRank according to the position of the target word. These four methods still need to be tested and evaluated. Intuitively, CRank(Middle) has a better performance as it simulates the most common cases for the target word being in the middle of the sentence. We also propose a special type of CRank(Single) that has no masks.

$$X'_i = \{\dots, W_{mask}, W_i, W_{mask}, \dots\} \tag{5}$$

$$Score(W_i) = Conf(X'_i) \tag{6}$$

Table 3. Scoring 'dumb' with classic and CRank. The original sentence is "it's dumb, but more importantly, it's just not scary".

Type	Sentense
Classic	it's [mask], but more importantly, it's just not scary.
CRank(Head)	dumb [mask] [mask] [mask] [mask] [mask] [mask]
CRank(Middle)	[mask] [mask] [mask] dumb [mask] [mask] [mask]
CRank(Tail)	[mask] [mask] [mask] [mask] [mask] [mask] dumb
CRank(Single)	dumb

4.1.3. CRankPlus

As our core concept of CRank involves reusing scores of words, we also consider taking the results of generating adversarial examples into account. If a word contributes to generating successful adversarial examples, we increase its score. Otherwise, we decrease it. Let the score of a word W be S, the new score be S' and the weight be α. Equation (7) shows our method and we normally set α below 0.05 to avoid a great rise or drop of the score.

$$S' = S \times (1 \pm \alpha) \tag{7}$$

4.2. Search Strategies

Search strategies mainly search through the ranked words and find a sequence of words that can generate a successful adversarial example. Two strategies are introduced in this section.

4.2.1. TopK

The TopK search strategy is mostly used in many well-known black box methods [7,8]. This strategy starts with the top word W_{R_1}, which has the highest score and increases one-by-one. As Equation (8) demonstrates, when processing a word W_{R_i}, we query the new sentence X'_i for its confidence. If the confidence satisfies Equation (9), we consider that the word is contributing toward generating an adversarial example, and keep it masked,

otherwise, we ignore the word. TopK continues until it masks the maximum allowed words or finds a successful adversarial example that satisfies Equation (1).

$$X'_i = \{\dots, W_{R_i-1}, W_{mask}, W_{R_i+1}, \dots\} \tag{8}$$

$$Conf(X'_i) < Conf(X'_{i-1}) \tag{9}$$

However, using the TopK search strategy breaks the connection between words. As Tables 2 and 4 demonstrates, when we delete the two words with the highest score, 'year' and 'taxes', its confidence is only 0.62. On the contrary, 'ex-wife' has the lowest score of 0.08, but it helps to generate an effective adversarial example when deleted with 'taxes'.

Table 4. Example of TopK. In this case, K is set to 2 and TopK fails to generate an adversarial example, while the successful one exists beyond the TopK search.

Label	Masked	Confidence	Status
TopK (Step 1)	taxes	0.71	Continue
TopK (Step 2)	taxes & year's	0.62	Reach K
Manual	taxes & ex-wife	0.49	Success

4.2.2. Greedy

To avoid the disadvantage of TopK and maintain an acceptable level of efficiency, we propose the greedy strategy. This strategy always masks the top-ranked word W_{R_1} as Equation (10) demonstrates, then uses word importance ranking to rank unmasked words again. It will continue until success or reaches the maximum amount of allowed words to be masked. However, the strategy only works with Classic WIR, not CRank.

$$X' = \{\dots, W_{R_1-1}, W_{mask}, W_{R_1+1}, \dots\} \tag{10}$$

4.3. Perturbation Methods

The major task of perturbation methods is making the target word deviated from the original position in the target model word vector space; thus, causing wrong predictions. Lin et al. [9] make a comprehensive summary of five perturbation methods: (1) insert a space or character into the word; (2) delete a letter; (3) swap adjacent letters; (4) Sub-C or replace a character with another one; (5) Sub-W or replace the word with a synonym. The first five are character-level strategies and the fifth is a word-level strategy. However, we innovate two new methods utilizing Unicode characters as Table 5 demonstrates. Sub-U randomly substitutes a character with a Unicode character that has a similar shape of meaning. Insert-U inserts a special Unicode character 'ZERO WIDTH SPACE', which is technically invisible in most text editors and printed papers, into the target word. Our methods have the same effectiveness as other character-level methods that turn the target word unknown to the target model. We do not discuss word-level methods as perturbation is not the focus of this paper.

Table 5. Our perturbation methods. The target model is CNN trained with SST-2. '$\wedge$' indicates the position of 'ZERO WIDTH SPACE'.

Method	Sentence	Prediction
	it 's dumb , but more importantly , it 's just not scary .	Negative (77%)
Sub-U	it 's dum B , but more importantly , it 's just not scαry .	Positive (62%)
Insert-U	it 's dum$\wedge$b , but more importantly , it 's just not sc$\wedge$ary .	Positive (62%)

5. Experiment and Evaluation

In this section, the setup of our experiment and the results are presented as follows.

5.1. Experiment Setup

Detailed information of the experiment, including datasets, pre-trained target models, benchmark, and the simulation environment are introduced in this section for the convenience of future research.

5.1.1. Datasets and Target Models

Three text classification tasks—SST-2, AG News, and IMDB—and two pre-trained models, word-level CNN and word-level LSTM from *TextAttack* [43], are used in the experiment. Table 6 demonstrates the performance of these models on different datasets.

Table 6. Accuracy of Target Models.

	SST-2	IMDB	AG News
CNN	82.68%	81%	90.8%
LSTM	84.52%	82%	91.9%

5.1.2. Implementation and Benchmark

We implement classic as our benchmark baseline. Our innovative methods are greedy, CRank, and CRankPlus. Each method will be tested in six sets of the experiment (two models on three datasets, respectively).

- Classic: classic WIR and TopK search strategy.
- Greedy: classic WIR and the greedy search strategy.
- CRank(Head): CRank-head and TopK search strategy.
- CRank(Middle): CRank-middle and TopK search strategy.
- CRank(Tail): CRank-tail and TopK search strategy.
- CRank(Single): CRank-single and TopK search strategy.
- CRankPlus: Improved CRank-middle and TopK search strategy.

5.1.3. Simulation Environment

The experiment is conducted on a server machine, whose operating system is Ubuntu 20.04, with 4 RTX 3090 GPU cards. TextAttack [43] framework is used for testing different methods. The first 1000 examples from the test set of each dataset are used for evaluation. When testing a model, if the model fails to predict an original example correctly, we skip this example. Three metrics in Table 7 are used to evaluate our methods.

Table 7. Evaluation Metrics.

Metric	Explanation
% Success	Successfully attacked examples/Attacked examples.
% Perturbed	Perturbed words/total words.
Query Number	Average queries for one successful adversarial example.

5.2. Performance

We analyze the effectiveness and the computational complexity of seven methods on the two models on three datasets as Table 8 demonstrates. In terms of the computational complexity, n is the word length of the attacked text. Classic needs to query each word in the target sentence and, thus, has a $O(n)$ complexity, while CRank uses a reusable query strategy and has a $O(1)$ complexity, as long as the test set is big enough. Moreover, our greedy has a $O(n^2)$ complexity, as with any other greedy search.

In terms of effectiveness, our baseline classic reaches a success rate of 67% at the cost of 102 queries, while CRank(Middle) reaches 66% with only 25 queries (increases 75%

efficiency, but has a 1% drop of the success rate, compared with classic). When we introduce greedy, it gains an 11% increase of the success rate, but consumes 2.5 times the queries. Among the sub-methods of CRank, CRank(Middle) has the best performance, so we refer to it as CRank in the following paper. As for CRankPlus, it has a very small improvement over CRank and we consider that it is because of our weak updating algorithm. For detailed results of the efficiency of all methods, see Figure 2; the distribution of the query number proves the advantage of CRank. In all, CRank proves its efficiency by greatly reducing the query number while keeping a similar success rate.

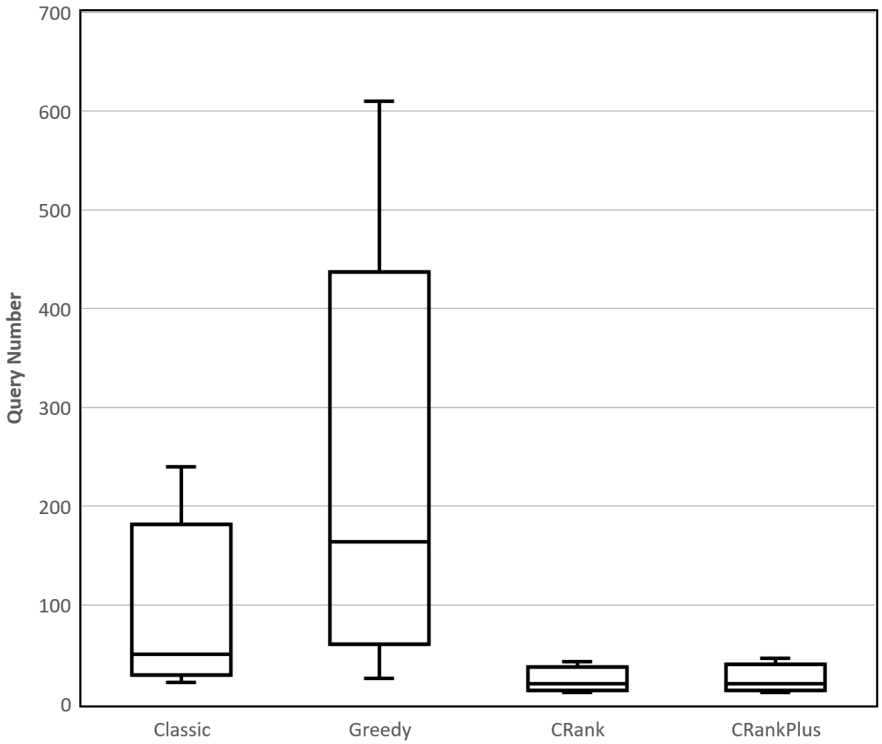

Figure 2. Query number distribution of classic, greedy, CRank, and CRankPlus.

Table 8. Average results. "QN" is query number. "CC" is computational complexity.

Method	% Sucess	% Perturbed	QN	CC
Classic	66.87	11.81	102	$O(n)$
Greedy	78.30	11.41	253	$O(n^2)$
CRank(Head)	63.36	12.90	28	$O(1)$
CRank(Middle)	65.91	12.76	25	$O(1)$
CRank(Tail)	64.79	12.60	26	$O(1)$
CRank(Single)	62.94	13.05	28	$O(1)$
CRankPlus	66.09	12.84	26	$O(1)$

In Table 9, we compare results of classic, greedy, CRank, and CRankPlus against CNN and LSTM. Despite greedy, all other methods have a similar success rate. However, LSTM is harder to attack and brings a roughly 10% drop in the success rate. The query number also rises with a small amount.

Table 9. Results of attacking various models. "QN" is query number.

Model	Method	% Success	% Perturbed	QN
CNN	Classic	71.83	12.42	99
	Greedy	84.30	11.76	238
	CRank	70.96	13.18	25
	CRankPlus	70.90	13.27	26
LSTM	Classic	61.91	11.21	105
	Greedy	72.29	11.05	268
	CRank	60.87	12.33	26
	CRankPlus	61.28	12.40	27

We also demonstrate the results of attacking various datasets in Table 10. Such results illustrate the advantages of CRank in two aspects. Firstly, when attacking datasets with very long text lengths, classic's query number grows linearly, while CRank keeps it small. Secondly, when attacking multi-classification datasets, such as AG News, CRank tends to be more effective than classic, as its success rate is 8% higher. Moreover, our innovated greedy achieves the highest success rate in all datasets, but consumes most queries.

Table 10. Results of attacking various datasets. "QN" is query number.

Dataset	Method	% Success	% Perturbed	QN
SST-2(17 [1])	Classic	75.92	17.73	23
	Greedy	80.94	16.33	27
	CRank	75.59	19.71	12
	CRankPlus	76	19.83	12
IMDB(266)	Classic	73.17	2.63	233
	Greedy	84.52	2.50	569
	CRank	62.79	2.87	43
	CRankPlus	62.57	3.02	46
AG News(38)	Classic	51.53	15.09	50
	Greedy	69.44	15.4	165
	CRank	59.37	15.69	21
	CRankPlus	59.7	15.66	21

[1] Average Text Length (words) of Attacked Examples.

5.3. Length of Masks

In this section, we analyze the influence of masks. As we previously pointed out, longer masks will not affect the effectiveness of CRank while shorter ones do. To prove our point, we designed an extra experiment that ran with Word-CNN on SST-2 and evaluated CRank-head, CRank-middle, and CRank-tail with different mask lengths. Among these methods, CRank-middle has double-sized masks because it has both masks before and after the word, as Table 3 demonstrates. Figure 3 shows the result that the success rate of each method tends to be stable when the mask length rises over four, while a shorter length brings instability. During our experiment of evaluating different methods, we set the mask length to 6 and it is reasonable.

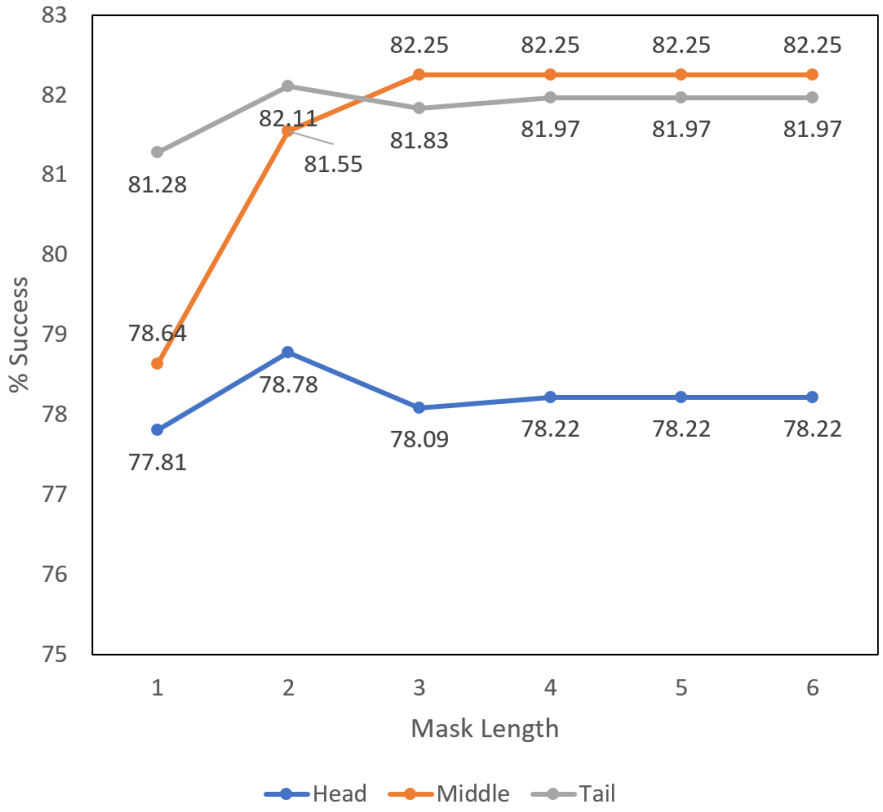

Figure 3. The Influence of mask length. The target model is CNN trained with SST-2.

6. Discussions

6.1. Word-Level Perturbations

In this paper, our attacks do not include word-level perturbations for two reasons. Firstly, the main focus of this paper is improving word importance ranking. Secondly, introducing word-level perturbations increases the difficulty of the experiment, which makes it unclear to express our idea. However, our three step attack can still adopt word-level perturbations in further work.

6.2. Greedy Search Strategy

Greedy is a supernumerary improvement for the text adversarial attack in this paper. In the experiment, we find that it helps to achieve a high success rate, but needs many queries. However, when attacking datasets with a short length, its efficiency is still acceptable. Moreover, if we are not sensitive about efficiency, greedy is a good choice for better performance.

6.3. Limitations of Proposed Study

In our work, CRank achieves the goal of improving the efficiency of the adversarial attack, yet there are still some limitations of the proposed study. Firstly, the experiment only includes text classification datasets and two pre-trained models. In further research, datasets of other NLP tasks and state-of-the-art models such as BERT [42] can be included. Secondly, CRankPlus has a very weak updating algorithm and needs to be optimized for better performance. Thirdly, CRank works under the assumption that the target model will returns confidence in its predictions, which limits its attacking targets.

6.4. Ethical Considerations

We present an efficient text adversarial method, CRank, mainly aimed at quickly exploring the shortness of neural network models in NLP. There is indeed a possibility

that our method is maliciously used to attack real applications. However, we argue that it is necessary to study these attacks openly if we want to defend them, similar to the development of the studies on cyber attacks and defenses. Moreover, the target models and datasets used in this paper are all open source and we do not attack any real-world applications.

7. Conclusions

In this paper, we firstly introduced a three-step adversarial attack for NLP models and presented CRank that greatly improved efficiency compared with classic methods. We evaluated our method and successfully improved efficiency by 75% at the cost of only a 1% drop of the success rate. We proposed the greedy search strategy and two new perturbation methods, Sub-U and Insert-U. However, our method needs to be improved. Firstly, in our experiment, the result of CRankPlus had little improvement over CRank. This suggests that there is still room for improvement with CRank concerning the concept of reusing previous results to generate adversarial examples. Secondly, we assume that the target model will return confidence in its predictions. The assumption is not realistic in real-world attacks, although many other methods are based on the same assumption. Thus, attacking in an extreme black box setting, where the target model only returns the prediction without confidence, is challenging (and interesting) for future work.

Author Contributions: Writing–original draft preparation, X.C.; writing—review and editing, B.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: The authors thank Fuqiang Lin for his suggestions in this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. In Proceedings of the 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, 14–16 April 2014.
2. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *Statistics* **2015**, *1050*, 20.
3. Kwon, H.; Kim, Y.; Yoon, H.; Choi, D. Random untargeted adversarial example on deep neural network. *Symmetry* **2018**, *10*, 738. [CrossRef]
4. Alzantot, M.; Sharma, Y.; Elgohary, A.; Ho, B.J.; Srivastava, M.; Chang, K.W. Generating Natural Language Adversarial Examples. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2890–2896.
5. Belinkov, Y.; Bisk, Y. Synthetic and Natural Noise Both Break Neural Machine Translation. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
6. Ebrahimi, J.; Lowd, D.; Dou, D. On Adversarial Examples for Character-Level Neural Machine Translation. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 653–663.
7. Gao, J.; Lanchantin, J.; Soffa, M.L.; Qi, Y. Black box generation of adversarial text sequences to evade deep learning classifiers. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24–24 May 2018; pp. 50–56.
8. Jin, D.; Jin, Z.; Zhou, J.T.; Szolovits, P. Is bert really robust? A strong baseline for natural language attack on text classification and entailment. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34.
9. Li, J.; Ji, S.; Du, T.; Li, B.; Wang, T. TextBugger: Generating Adversarial Text Against Real-world Applications. In Proceedings of the 26th Annual Network and Distributed System Security Symposium, San Diego, CA, USA, 24–27 February 2019.
10. Liang, B.; Li, H.; Su, M.; Bian, P.; Li, X.; Shi, W. Deep Text Classification Can be Fooled. In Proceedings of the IJCAI, Stockholm, Sweden, 13–19 July 2018.
11. Li, L.; Ma, R.; Guo, Q.; Xue, X.; Qiu, X. BERT-ATTACK: Adversarial Attack against BERT Using BERT. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 6193–6202.

12. Zivkovic, M.; Bacanin, N.; Venkatachalam, K.; Nayyar, A.; Djordjevic, A.; Strumberger, I.; Al-Turjman, F. COVID-19 cases prediction by using hybrid machine learning and beetle antennae search approach. *Sustain. Cities Soc.* **2021**, *66*, 102669. [CrossRef] [PubMed]
13. Cao, J.; Gao, J.; Rad, H.N.; Mohammed, A.S.; Hasanipanah, M.; Zhou, J. A novel systematic and evolved approach based on XGBoost-firefly algorithm to predict Young's modulus and unconfined compressive strength of rock. *Eng. Comput.* **2021**. [CrossRef]
14. Selvakumar, B.; Muneeswaran, K. Firefly algorithm based feature selection for network intrusion detection. *Comput. Secur.* **2019**, *81*, 148–155.
15. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [CrossRef]
16. Han, G.; Huang, S.; Ma, J.; He, Y.; Chang, S.F. Meta Faster R-CNN: Towards Accurate Few-Shot Object Detection with Attentive Feature Alignment. *arXiv* **2021**, arXiv:2104.07719.
17. Vedantam, V.K. The Survey: Advances in Natural Language Processing using Deep Learning. *Turk. J. Comput. Math. Educ. (TURCOMAT)* **2021**, *12*, 1035–1040. [CrossRef]
18. Alberti, C.; Ling, J.; Collins, M.; Reitter, D. Fusion of Detected Objects in Text for Visual Question Answering. In Proceedings of the EMNLP/IJCNLP, Hong Kong, China, 3–7 November 2019.
19. Baevski, A.; Edunov, S.; Liu, Y.; Zettlemoyer, L.; Auli, M. Cloze-driven Pretraining of Self-attention Networks. In Proceedings of the EMNLP/IJCNLP, Hong Kong, China, 3–7 November 2019.
20. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2574–2582.
21. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017; pp. 39–57.
22. Jia, R.; Liang, P. Adversarial Examples for Evaluating Reading Comprehension Systems. In Proceedings of the EMNLP, Copenhagen, Denmark, 7–11 September 2017.
23. Wang, Y.; Bansal, M. 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, New Orleans, LA, USA, 1–6 June 2018; Volume 2, pp. 575–581.
24. Ebrahimi, J.; Rao, A.; Lowd, D.; Dou, D. HotFlip: White-Box Adversarial Examples for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; Volume 2, pp. 31–36.
25. Garg, S.; Ramakrishnan, G. BAE: BERT-based Adversarial Examples for Text Classification. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 6174–6181.
26. Iyyer, M.; Wieting, J.; Gimpel, K.; Zettlemoyer, L. 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, New Orleans, LA, USA, 1–6 June 2018; Volume 1, pp. 1875–1885.
27. Wang, B.; Pei, H.; Pan, B.; Chen, Q.; Wang, S.; Li, B. T3: Tree-Autoencoder Regularized Adversarial Text Generation for Targeted Attack. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 6134–6150.
28. Wallace, E.; Stern, M.; Song, D. Imitation Attacks and Defenses for black box Machine Translation Systems. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 5531–5546.
29. Ren, S.; Deng, Y.; He, K.; Che, W. Generating natural language adversarial examples through probability weighted word saliency. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1085–1097.
30. Pennington, J.; Socher, R.; Manning, C.D. Glove: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
31. Kwon, H. Friend-Guard Textfooler Attack on Text Classification System. *IEEE Access* **2021**. [CrossRef]
32. Grosse, K.; Papernot, N.; Manoharan, P.; Backes, M.; McDaniel, P. Adversarial examples for malware detection. In Proceedings of the European Symposium on Research in Computer Security, Oslo, Norway, 11–15 September 2017; pp. 62–79.
33. Suciu, O.; Coull, S.E.; Johns, J. Exploring adversarial examples in malware detection. In Proceedings of the 2019 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 19–23 May 2019; pp. 8–14.
34. Alasmary, H.; Abusnaina, A.; Jang, R.; Abuhamad, M.; Anwar, A.; Nyang, D.; Mohaisen, D. Soteria: Detecting adversarial examples in control flow graph-based malware classifiers. In Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), Singapore, 29 November–1 December 2020; pp. 888–898.
35. Yang, K.; Liu, J.; Zhang, C.; Fang, Y. Adversarial examples against the deep learning based network intrusion detection systems. In Proceedings of the MILCOM 2018—2018 IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, 29–31 October 2018; pp. 559–564.

36. Hashemi, M.J.; Keller, E. Enhancing robustness against adversarial examples in network intrusion detection systems. In Proceedings of the 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Leganes, Spain, 10–12 November 2020; pp. 37–43.
37. Athalye, A.; Carlini, N.; Wagner, D. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 274–283.
38. Wang, T.; Wang, X.; Qin, Y.; Packer, B.; Li, K.; Chen, J.; Beutel, A.; Chi, E. CAT-Gen: Improving Robustness in NLP Models via Controlled Adversarial Text Generation. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online, 16–20 November 2020; pp. 5141–5146.
39. Liu, H.; Zhang, Y.; Wang, Y.; Lin, Z.; Chen, Y. Joint character-level word embedding and adversarial stability training to defend adversarial text. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 8384–8391.
40. Wang, X.; Yang, Y.; Deng, Y.; He, K. Adversarial Training with Fast Gradient Projection Method against Synonym Substitution Based Text Attacks. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 2–9 February 2021; Volume 35, pp. 13997–14005.
41. Gan, Y.; Chen, X.; Huang, Q.; Purver, M.; Woodward, J.R.; Xie, J.; Huang, P. Towards Robustness of Text-to-SQL Models against Synonym Substitution. In Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, Virtual Event, 1–6 August 2021; Volume 1, pp. 2505–2515.
42. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL-HLT, Minneapolis, MN, USA, 2–7 June 2019.
43. Morris, J.; Lifland, E.; Yoo, J.Y.; Grigsby, J.; Jin, D.; Qi, Y. TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 16–20 November 2020; pp. 119–126.