

國立臺灣大學電機資訊學院電機工程學系



碩士論文

Department of Electrical Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

防禦針對文字分類模型的對抗性攻擊

Defend against adversarial attacks in text classification

林俊燁

Chun-Yeh Lin

指導教授: 雷欽隆 博士

Advisor: Chin-Laung Lei Ph.D.

中華民國 112 年 1 月

January, 2023

國立臺灣大學碩士學位論文

口試委員會審定書



防禦針對文字分類模型的對抗性攻擊

Defend against adversarial attacks in text
classification

本論文係林俊燁君 (R09921093) 在國立臺灣大學電機工程學系完成之碩士學位論文，於民國 112 年 1 月 12 日承下列考試委員審查通過及口試及格，特此證明

口試委員： 雷欽隆

(指導教授)

顏麗娟 _____

紀博文 _____

所長： 李建橫





Acknowledgements

非常榮幸有這個機會進入台大電機所碩士班就讀，在求學期間都能與最頂尖的學生們一起學習，並在彼此討論中對知識及技術有更深刻的理解。

特別謝謝雷欽隆老師，在研究上給予最大的協助，讓我對於需要更深入探討的部分或是需要調整的方向，都能夠盡快做出修正，最終得以完成自己的研究及論文。謝謝實驗室同學庭豪、政剛、翔予以及翊維，在修課及研究討論中，學到了許多新的知識及工具。謝謝家慶學長以及淑萍學姊，熱心處理實驗室事務，給予我們很大的協助及溫暖。





摘要

近年來深度學習在自然語言處理的問題上取得了卓越的成果，然而日常生活中常見的應用，例如垃圾訊息過濾以及情緒分析等等，都很容易受到對抗性攻擊，導致安全性上的疑慮。

本文提出兩個方法，干擾偵測可以判斷文字是否受到字元修改的攻擊，並接著基於上下文將受到修改的文字恢復成可能的替代字詞。在字詞替換攻擊上，藉由將重要的字詞替換成數個可能的替代文字以增加樣本數量，且預測結果為所有增加的樣本中最多數被分到的類別。

本文提出的方法可以在不需要知道模型參數以及調整模型架構的條件下抵禦對抗式攻擊。在 IMDb 資料集上所完成的實驗證明，本文的方法可以有效防禦在文字分類上的字元替換及字詞替換攻擊，並展現比比較基準更好的成果。

關鍵字：防禦對抗性攻擊、文字分類





Abstract

In recent years, deep learning models have achieved prominent success on NLP tasks. However, widely used real-world applications such as spam filter and sentiment analysis are vulnerable to adversarial attacks.

This thesis proposes two methods to defend against adversarial attacks on the sentiment analysis task. Perturbation detector detects if a token in the sample is perturbed through character level attacks, and the recovery process recovers the words from the perturbed ones to possible substitutions based on the context. For word level attacks, augmenting inputs by replacing important words to their possible substitutions and the result of the original sample is the majority class among all the augmented samples.

Our methods can block adversarial attacks without knowing the model parameters and modifying model structures. Experiments on IMDB dataset demonstrate that our methods can effectively block both character level and word level attacks and outperform baseline method on text classification task.

Keywords: adversarial attack defense, text classification





Contents

	Page
Verification Letter from the Oral Examination Committee	i
Acknowledgements	iii
摘要	v
Abstract	vii
Contents	ix
List of Figures	xi
List of Tables	xiii
Chapter 1 Introduction	1
Chapter 2 Related Work	3
2.1 TextBugger	3
2.2 TextFooler	5
2.3 Discriminate Perturbations	7
Chapter 3 Methodology	9
3.1 Character level defense	9
3.1.1 Perturbation Detector	9
3.1.2 Perturbation Recovery	11
3.2 Word level defense	12
3.2.1 Important word replacing	13

3.3	Overall attacks defense	14
Chapter 4	Experiments	17
4.1	Experiment settings	17
4.1.1	Dataset	17
4.1.2	Adversarial attacks	17
4.1.3	Base model and baseline	19
4.1.4	Evaluation metric	19
4.2	Experimental results	20
4.2.1	Performance of perturbation detector	20
4.2.2	Effectiveness of augmenting input	21
4.2.3	Defend against overall attacks	22
Chapter 5	Conclusion	23
	References	25





List of Figures

2.1	Process of adversarial data augmentation	4
2.2	Algorithm of TextFooler	6
2.3	Schema of DISP framework	7
3.1	Word embedding distribution of overdone in the original sentence "Everything was overdone to the point of absurdity."	10
3.2	Word embedding distribution of overdone in the perturbed sentence "Everything was overdone to the point of absurdity."	11
3.3	Structure to defend against overall attacks	15





List of Tables

2.1	Examples of perturbations generated by TextBugger	4
3.1	Example of perturbation detector and recovery	12
3.2	Example of augmented samples for one masked word	14
4.1	Examples of character level attacks	18
4.2	Examples of word substitutions	18
4.3	Limitations of spelling check	20
4.4	The accuracy of methods on character level attacks	21
4.5	The accuracy of methods on word level attack	22
4.6	The accuracy of methods on overall attacks	22





Chapter 1 Introduction

Deep neural networks have achieved prominent results on Natural Language Processing (NLP) tasks in recent years. Although such DNN models are widely applied to research and commercial systems, they are vulnerable to adversarial examples which are intentionally generated by attackers. Many studies have proved that adding perturbations to benign inputs could fool the targeted models. Such discovery has raised serious concerns on the security issues of DNN NLP tasks.

Common adversarial examples are generated by inserting, swapping, replacing characters [5] [4] or substituting words with their synonyms[6] [10][16]. These examples aim to conserve semantic similarity with original samples and not be aware of humans but be able to confuse deep neural network models without considering possible adversaries.

DNN-based classification tasks play an important role in information understanding and analyzing. For instance, many recommendation systems rely on sentiment analysis to deliver suitable results. Adversarial attacks on such classifiers can lead to incorrect predictions and cause unexpected outcomes.

It is not straight forward to adapt existing approaches such as data augmentation [8] and adversarial training [13] used in image domain to NLP tasks. Images have continuous pixels while text are discrete tokens. Therefore, defending against adversarial attacks in

NLP tasks remains a challenging and unsolved problem.

In this thesis, we proposed two methods to defend against adversarial attacks on the sentiment analysis task. Perturbation detector detects if a token is modified through character level perturbation and then recovers the perturbed token to possible words based on the context. For word level attacks, we augment an input by replacing important words with their possible substitutions. The result class of the input is the majority class among all augmented samples predicted by the model.

Our methods can block both character level and word level attacks in text classification tasks. In addition, the prior knowledge to parameters of the classification model and the access to retrain or modify the model structure is not necessary.





Chapter 2 Related Work

2.1 TextBugger

TextBugger [9] proposed 5 methods to perturb the target words which are (1) Insert : insert a space into a word, (2) Delete : delete a character of the word except the first and the last ones, (3) Swap : swap two adjacent characters except the first and the last ones, (4) Substitute-C : replace characters with virtually similar ones and (5) Substitute-W : replace the word with synonyms in the embedding vector [15]. TextBugger achieved successful attacks on both white-box and black box scenarios.

Experiment conducted in the study also shows that spelling check and adversarial training using adversarial samples generated by TextBugger are effective in defending the attack.

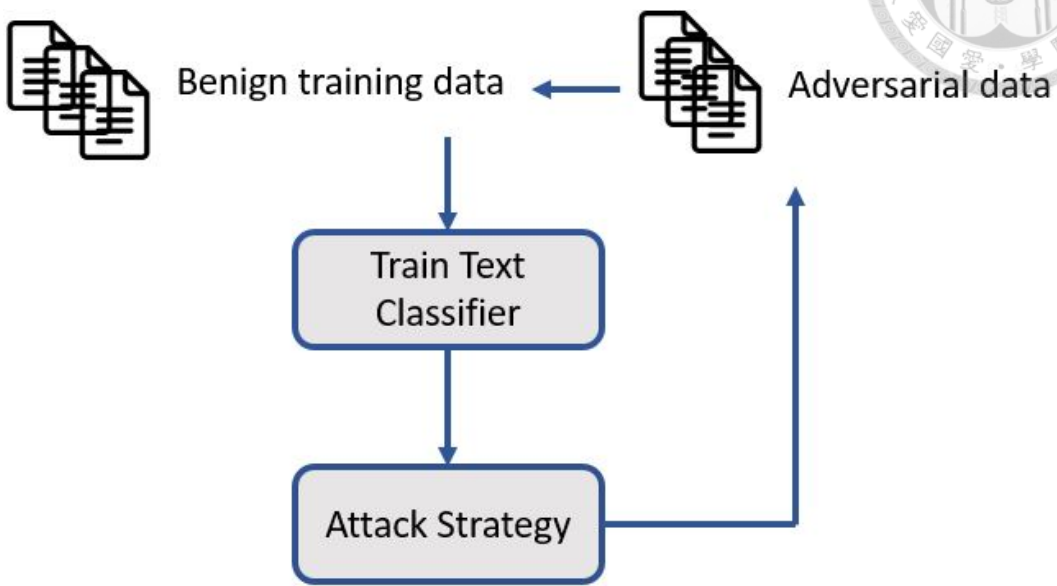


Figure 2.1: Process of adversarial data augmentation

Table 2.1: Examples of perturbations generated by TextBugger

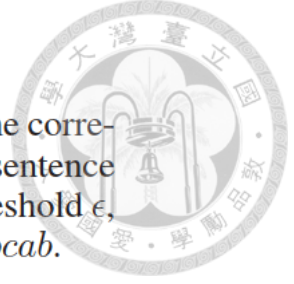
Original	Insert	Delete	Swap	Substitute - C	Substitute - W
good	g ood	god	good	go0d	great
awesome	aw esome	awsome	awesmoe	awesone	amazing

2.2 TextFooler



TextFooler [7] is a typical word level attack, which replaces the target words with synonyms extracted from word embedding [14]. The result shows that not only simple CNN and RNN models are vulnerable to the attack but also pre-trained BERT model is not robust to synonym substitutions. Further, the result of human evaluation reveals that synonym substitution is hardly recognize by human.

In this study, besides the success of the attack strategy, adversarial training using adversarial samples generated by TextFooler could decrease the success rate of the attack and increase the perturbation percentage required to evade target models.



Input: Sentence example $X = \{w_1, w_2, \dots, w_n\}$, the corresponding ground truth label Y , target model F , sentence similarity function Sim , sentence similarity threshold ϵ , word embeddings Emb over the vocabulary $Vocab$.

Output: Adversarial example X_{adv}

```

1: Initialization:  $X_{adv} \leftarrow X$ 
2: for each word  $w_i$  in  $X$  do
3:   Compute the importance score  $I_{w_i}$  via Eq.2
4: end for
5:
6: Create a set  $W$  of all words  $w_i \in X$  sorted by the descending order of their importance score  $I_{w_i}$ .
7: Filter out the stop words in  $W$ .
8: for each word  $w_j$  in  $W$  do
9:   Initiate the set of candidates CANDIDATES by extracting the top  $N$  synonyms using  $CosSim(Emb_{w_j}, Emb_{word})$  for each word in  $Vocab$ .
10:  CANDIDATES  $\leftarrow$  POSFilter(CANDIDATES)
11:  FINCANDIDATES  $\leftarrow$  { }
12:  for  $c_k$  in CANDIDATES do
13:     $X' \leftarrow$  Replace  $w_j$  with  $c_k$  in  $X_{adv}$ 
14:    if  $Sim(X', X_{adv}) > \epsilon$  then
15:      Add  $c_k$  to the set FINCANDIDATES
16:       $Y_k \leftarrow F(X')$ 
17:       $P_k \leftarrow F_{Y_k}(X')$ 
18:    end if
19:  end for
20:  if there exists  $c_k$  whose prediction result  $Y_k \neq Y$  then
21:    In FINCANDIDATES, only keep the candidates  $c_k$  whose prediction result  $Y_k \neq Y$ 
22:     $c^* \leftarrow \underset{c \in FINCANDIDATES}{\operatorname{argmax}} Sim(X, X'_{w_j \rightarrow c})$ 
23:     $X_{adv} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{adv}$ 
24:    return  $X_{adv}$ 
25:  else if  $P_{Y_k}(X_{adv}) > \min_{c_k \in FINCANDIDATES} P_k$  then
26:     $c^* \leftarrow \underset{c_k \in FINCANDIDATES}{\operatorname{argmin}} P_k$ 
27:     $X_{adv} \leftarrow$  Replace  $w_j$  with  $c^*$  in  $X_{adv}$ 
28:  end if
29: end for
30: return None

```

Figure 2.2: Algorithm of TextFooler



2.3 Discriminate Perturbations

This study [17] uses two main modules to defend against adversarial samples. Perturbation Discriminator can detect if any token in an input sample is perturbed, which is trained using adversarial samples as data. After determining the perturbed tokens, Embedding Estimator comes along to estimate the original embedding of the perturbed tokens. Therefore, the process recovers the adversarial samples and then forward the samples into models to deliver correct predictions.

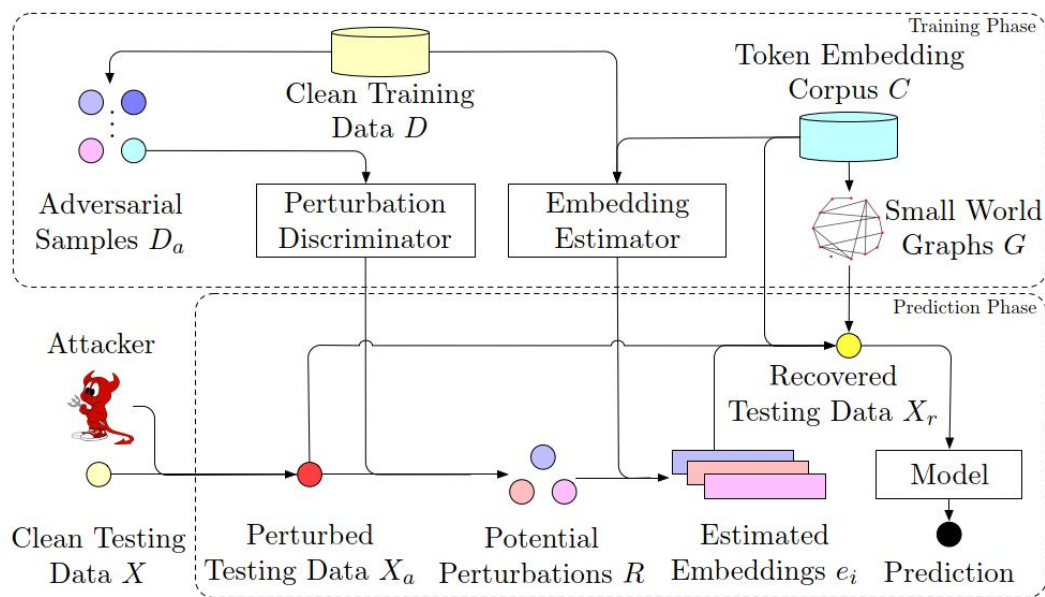


Figure 2.3: Schema of DISP framework





Chapter 3 Methodology

3.1 Character level defense

Character level attacks usually modify the target words by inserting, deleting, swapping, replacing characters in the word, which aim to map the embedding of the original words to unknown words, therefore, affecting the impact of the words on the model prediction. To defend against character level attacks, we adopt a perturbation detector to predict if a word is modified with character perturbations. When a word is predicted as a potential perturbed word, the recovery process takes place to recover the word from perturbed to the possible substitution based on the context.

3.1.1 Perturbation Detector

The character perturbation detector is a classifier build to detect if a token t_i in an input S is perturbed based on the context. We first use BERT[3] tokenizer and model to derive contextualized word embedding E_i for each token t_i and then cascade it with a logistic regression classifier to predict if the token t_i is perturbed. Each contextual embedding of contains the dimension of 768. Figure 3.1 and 3.2 shows the distribution of the original word embedding and the perturbed word embedding. The classifier takes contex-

tualized word embedding E_i of the token t_i as an input and output two classes $\{0, 1\}$ to determine whether the word is perturbed.

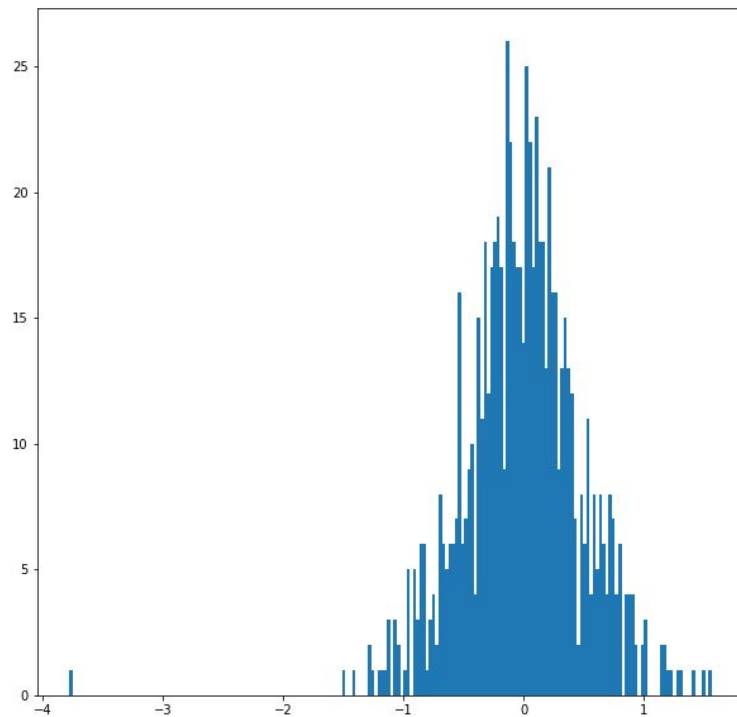


Figure 3.1: Word embedding distribution of overdone in the original sentence "Everything was overdone to the point of absurdity."

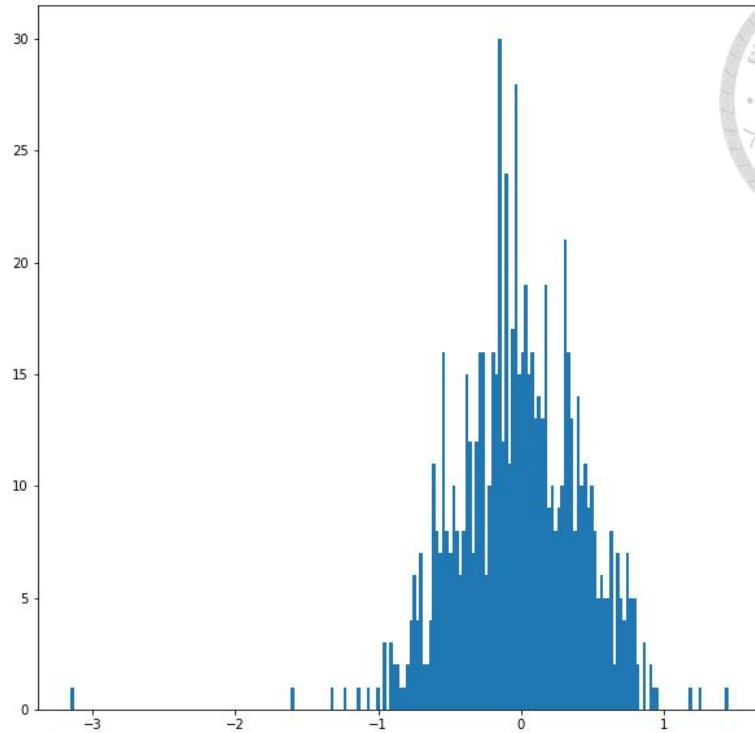


Figure 3.2: Word embedding distribution of over d_0 ne in the perturbed sentence "Everything was over d_0 ne to the point of absurdity."

3.1.2 Perturbation Recovery

Here we utilized BERT masked language model[11] to recover the perturbed tokens. When a token t_i is predicted as an potential perturbation, the sentence containing the token is selected as the context for estimating the possible original word. For a perturbed token t_i , the token is replaced with the <mask> token. BERT-MLM will predict the possible words to fill in the masked position. We select the most probable one as the substitution to replace the perturbed token.



Table 3.1: Example of perturbation detector and recovery

Original	The acting seems very unrealistic and is generally poor.
Perturbed	The acting seems very unrealistic and is generally poor .
Masked	The acting seems very unrealistic and is generally poor.
Recovered	The acting seems very unrealistic and is generally boring .

3.2 Word level defense

Word level attacks usually replace the target word with their synonyms to preserve semantic similarity, which decrease awareness by human and try to fool the model in the meantime. Adversarial samples modified through word level perturbations are hardly recognized by human and detected by the model trained based on the word embedding.

According to the common approaches that most attack strategies choose important words in an input to modify, we assume that important words in an input sample are modified by attackers. Therefore, we generate augmented inputs by replacing important words with possible words at the same position based on the context, aiming to smooth the extreme impact on the intentionally perturbed word.



3.2.1 Important word replacing

We first define the score I_{w_i} representing the importance of a word in a sample $s = [w_0 .. w_n]$. We denote $X_{/w_i}$ as a input $X = [w_0 .. w_n]$ removing a word w_i , that is, $X_{/w_i} = [w_0 .. w_{i-1} w_{i+1} .. w_n]$. The importance score is calculated by the prediction difference before and after deleting the word w_i .

Each token in a sample is sorted by the importance then we mask the word by the order of importance and use the BERT masked language model to predict the top 5 most probable words to fill the position of the masked word. These five new samples are appended to a set. Table 3.2 shows the augmented inputs generated through the process. Each sample produced last round loops to mask the next important word.

In particular, there are 5 augmented samples at the end of first loop, 25 for the second and 125 for the last round. Therefore, there are 155 augmented samples at the end of the whole process. The final classification result will be the most probable class among all these augmented samples in the set. For example, if the prediction result are 120 positive and 35 negative among 155 augmented samples in the set, the prediction result of the original input will be delivered as positive.

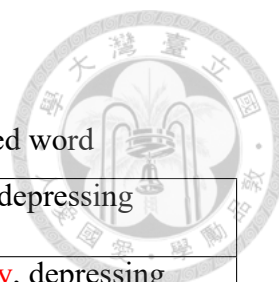


Table 3.2: Example of augmented samples for one masked word

Original	A rating of "1" does not begin to express how dull, depressing and relentlessly bad this movie is.
Perturbed	A rating of "1" does not begin to express how dreary , depressing and relentlessly bad this movie is.
Masked	A rating of "1" does not begin to express how <mask>, depressing and relentlessly bad this movie is.
Augmented	<p>A rating of "1" does not begin to express how awful, depressing and relentlessly bad this movie is.</p> <p>A rating of "1" does not begin to express how gross, depressing and relentlessly bad this movie is.</p> <p>A rating of "1" does not begin to express how depressing, depressing and relentlessly bad this movie is.</p> <p>A rating of "1" does not begin to express how bleak, depressing and relentlessly bad this movie is.</p> <p>A rating of "1" does not begin to express how horrible, depressing and relentlessly bad this movie is.</p>

3.3 Overall attacks defense

Overall attacks combines both character level and word level attacks, which is more likely to be applied by attackers targeting real-world applications. We concatenate methods proposed in the above sections to defend against overall attacks. Input sequence first goes through the perturbation detector. if there is any perturbed token, the token is passed to recovery process. Important word replacement is followed by the perturbation detector to augment the input. Augmented inputs are passed to the classifier and deliver the final result at the end. Figure 3.3 is the structure of our methods to defend against overall attacks.

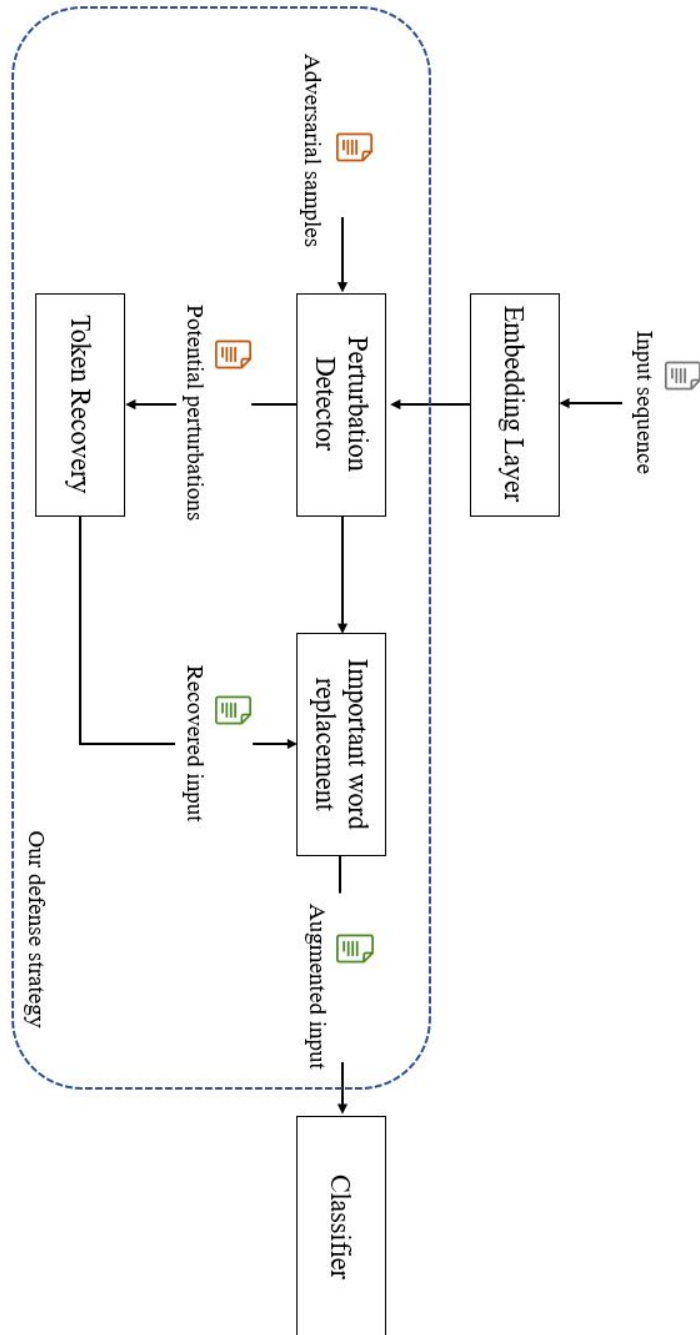


Figure 3.3: Structure to defend against overall attacks





Chapter 4 Experiments

4.1 Experiment settings

4.1.1 Dataset

The experiment is conducted on the dataset Internet Movie Database (IMDb) [1], which is a sentiment classification dataset containing movie reviews labeled with positive and negative sentiment classes. The dataset has 25,000 training data and 25,000 testing data.

4.1.2 Adversarial attacks

We consider four types of character level attacks, which contains Insertion, Deletion, Swap and Substitute. Insertion inserts a random character or the same character to a random position of the word. Deletion deletes a random character in the word. Swap flips two adjacent characters in the word. Substitute replaces a random character in the word with a similar looking character (e.g., replacing “o” with “0”, ”p” with ”q”, “l” with “1”, “a” with “@”) or adjacent character on the keyboard (e.g., replacing “m” with “n”, ”s” with ”d”, ”k” with ”l”). Table 4.1 shows the examples of character level attacks implemented in this study.



Table 4.1: Examples of character level attacks

Original	outstanding	well	excellent	dull
Insert	outstaanding	welll	excellent	duell
Swap	otustanding	wlel	excellent	dlul
Delete	outstading	wel	excelent	dll
Substitute	outst@nding	well	excllemt	dill

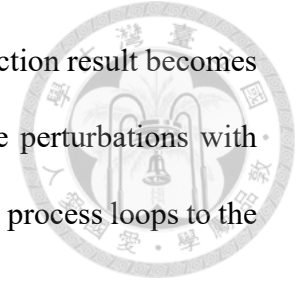
For word level attack, we use pre-trained GloVe word vector [15] to substitute target words with their similar words, which are the nearest words to the target word in the word vector space. Top five similar words are selected as alternatives to perturb the sample. Table 4.2 shows the examples of substitutions to replace target words.

Table 4.2: Examples of word substitutions

Target	outstanding	well	excellent	dull
Similar Substitutions	exceptional	good	fantastic	bore
	phenomenal	sure	great	boring
	terrific	yes	brilliant	rubbish
	fantastic	yeah	superb	gloomy
	superb	better	outstanding	messy

Perturbations are added to the top five important words by the order of their importance. To preserve semantic similarity we use Universal Sentence Encoder[2] which calculates similarity between two sentences. The constraint of the similarity between original and perturbed sentence is set as 0.7.

The adversarial sample generating process stops when the prediction result becomes different to the predicted label without perturbations, otherwise the perturbations with largest prediction difference toward the other class is selected and the process loops to the next target word.



4.1.3 Base model and baseline

We consider the LSTM model as our base model to evaluate vulnerability without defense and the performance of defense methods. To evaluate our defense method, we implemented the following baseline : (1) Spelling check[12], which is proved effective defending against text adversarial attack.[9] (2) The original DISP model[17]. Baseline methods as well as our methods are independent to the prediction model. Knowledge of model parameters and the access to retrain the model are not necessary.

4.1.4 Evaluation metric

We evaluate the defense performance by classification accuracy. The accuracy increase of a defense method against adversarial attacks is equivalent to the effectiveness of the defense strategy.



4.2 Experimental results

4.2.1 Performance of perturbation detector

Table 4.4 shows the accuracy of all character level defense methods. After we applied character level attacks to the model with original accuracy 88.7%, the accuracy drops to 68.1%. According to the results, baselines and our methods are effective defending against character level attacks. Spelling check achieved the accuracy of 73.1%, which is the least effective method among all implemented methods. Moreover, we discovered that spelling check suffers from recognizing perturbations which substitute alphabetic characters with non-alphabetic characters. Spelling check has limitations detecting uncommon and intentionally generated typos as well as correcting words based on the context. Table 4.3 shows the limitations of spelling check.

Table 4.3: Limitations of spelling check

Original	Perturbed	Recovered
well	well	well
garage	g@rage	g@rage
film	f!lm	f!lm
dull	dill	will

Compared to DISP recovered the model accuracy from 68.1% to 77.4%, our method achieved 81.6% accuracy after applying defense strategy. The DISP model increase the accuracy by 9.3% and our perturbation detector by 13.5%, which performs better than the DISP model on the character level attacks.

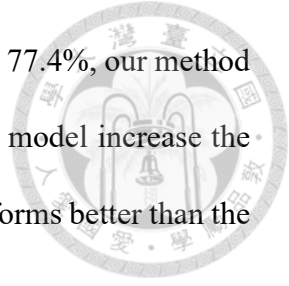


Table 4.4: The accuracy of methods on character level attacks

Method	Attack-free	Character level attacks
Original LSTM	88.7%	68.1%
Spelling check		73.1%
DISP		77.4%
Perturbation detector		81.6%

4.2.2 Effectiveness of augmenting input

Table 4.5 shows the experiment results on defending against the word level attack. The original model accuracy is 88.7% on the data without perturbations. The accuracy drops to 76.2% after we generate word level adversarial samples from the the data. Spelling check does not show significant effectiveness defending against word level attack. Moreover, the accuracy even drops a little to 75.8%, which might caused by false positive detection.

Both DISP model and our method increase the model accuracy when the defense strategies are applied. Compared to the DISP model, the accuracy is 79.3% and our method is 80.1%, which recover the accuracy by 3.1% and 3.9%. Our method achieves better performance defending against the word level attack.



Table 4.5: The accuracy of methods on word level attack

Method	Attack-free	Word level attack
Original LSTM	88.7%	76.2%
Spelling check		75.8%
DISP		79.3%
Augmented input		80.1%

4.2.3 Defend against overall attacks

We applied adversarial attacks mixed with both character level and word level attacks mentioned in the previous sections to the LSTM model. Perturbation detector and augmenting input are combined to defend against the overall attacks. Table 4.4 shows the experiment result of baseline methods and our methods defending against overall adversarial attacks. The accuracy of original model is 88.7% and the accuracy drops to 61.7%. Spelling check performs least effective among all the methods. DISP recovers the accuracy from 61.7% to 74.6%, while our combined method achieves 77.5%. Our methods perform better than baseline methods facing adversarial attacks mixed with character level and word level perturbations.

Table 4.6: The accuracy of methods on overall attacks

Method	Attack-free	Character level attacks
Original LSTM	88.7%	61.7%
Spelling check		67.2%
DISP		74.6%
Perturbation detector + Augmented input		77.5%



Chapter 5 Conclusion

This thesis proposed defense methods to defend against adversarial attacks in text classification tasks. According to the experiment results, we can increase the model accuracy under both character level and word level attacks. Furthermore, this method can increase robustness without prior knowledge about actual classification model and the access to retrain the model.

Once NLP tasks and services achieve more remarkable outcomes, concerns on vulnerabilities and security issues of these models will also arise at the same time. Though not all the defense methods against attacks on images with outstanding results can be applied to the NLP domains, there have been some methods proved effective to attacks on text classification tasks. However, word level attacks is still hard to detect and discover both by human and computers. Defense methods against word level attacks on text can be further explored.

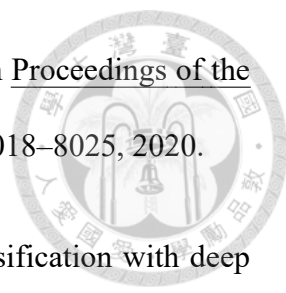




References

- [1] Imdb sentiment analysis dataset. <https://www.imdb.com/interfaces/>.
- [2] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder. [arXiv preprint arXiv:1803.11175](#), 2018.
- [3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. [arXiv preprint arXiv:1810.04805](#), 2018.
- [4] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou. Hotflip: White-box adversarial examples for text classification. [arXiv preprint arXiv:1712.06751](#), 2017.
- [5] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. In [2018 IEEE Security and Privacy Workshops \(SPW\)](#), pages 50–56. IEEE, 2018.
- [6] S. Garg and G. Ramakrishnan. Bae: Bert-based adversarial examples for text classification. [arXiv preprint arXiv:2004.01970](#), 2020.
- [7] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. 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, volume 34, pages 8018–8025, 2020.

- 
- [8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. Commun. ACM, 60(6):84–90, may 2017.
- [9] J. Li, S. Ji, T. Du, B. Li, and T. Wang. Textbugger: Generating adversarial text against real-world applications. arXiv preprint arXiv:1812.05271, 2018.
- [10] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu. Bert-attack: Adversarial attack against bert using bert. arXiv preprint arXiv:2004.09984, 2020.
- [11] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.
- [12] S. Loria. textblob documentation. Release 0.15, 2, 2018.
- [13] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017.
- [14] N. Mrkšić, D. O. Séaghdha, B. Thomson, M. Gašić, L. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen, and S. Young. Counter-fitting word vectors to linguistic constraints. arXiv preprint arXiv:1603.00892, 2016.
- [15] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543, 2014.
- [16] L. Sun, K. Hashimoto, W. Yin, A. Asai, J. Li, P. Yu, and C. Xiong. Adv-bert: Bert

is not robust on misspellings! generating nature adversarial samples on bert. [arXiv preprint arXiv:2003.04985](#), 2020.

- [17] Y. Zhou, J.-Y. Jiang, K.-W. Chang, and W. Wang. Learning to discriminate perturbations for blocking adversarial attacks in text classification. [arXiv preprint arXiv:1909.03084](#), 2019.

