Improving the Quality of Adversarial Examples via Contrastive Learning and Pretraining

Final Year Project First Term Report

by

Yung-chieh Huang

Supervised by

Professor Michael Rung-Tsong Lyu

Department of Computer Science and Engineering

The Chinese University of Hong Kong

Contents

1 Introduction 1

1.1 Adversarial Attack 1

1.2 Motivation 3

1.3 Objective 4

1.4 Contribution 5

2 Related Work 6

2.1 Adversarial Attack for Text 6

2.2 Pretraining 7

2.3 Contrastive Learning 7

3 Methedology 9

4 Experiments 11

4.1 Datasets 11

4.2 Set up 11

4.3 Results 12

4.3.1 Pretraining Only 13

4.3.2 Contrastive Learning and Pretraining 14

4.3.3 Using CLINE to Create Contrastive Sentences 16

5 Conclusion and Future Work 18

Chapter 1

Introduction

Deep learning has gained great success in the past few years. From image recognition to virtual assistant, it is undeniable that deep learning has made our lives more convenient than ever. Deep learning has also greatly influenced the field of natural language processing (NLP), brought about many new applications such as machine translation. Yet recently, researchers have found that several seemingly robust language models are vulnerable to adversarial attacks.

1.1 Adversarial Attack

Adversarial attack is an approach to test the robustness of machine learning models. An adversarial example is created by intentionally apply perturbations to an original sentence such that a victim model correctly classifies the original sentence but misclassifies the adversarial example with high confidence. A well-crafted adversarial example should have minimum perturbations, it should also preserve the structure and characteristics of the original and be as close to the original sentence as possible.

The field of adversarial attack has been well explored in image recognition, yet only until recently do researchers start looking into adversarial attack in NLP. This is due to the challenges in generating text adversarial examples. Image adversarial examples can be created by simply perturbing pixels, those changes are hardly distinguishable to human perception but are effective in fooling machine learning models (Szegedy et al. 2013; Goodfellow, Shlens, and Szegedy 2014). On the other hand, altering a word in a sentence can change the whole semantic drastically. Moreover, words are discrete, so gradient descent is not applicable. This project focuses on NLP adversarial attack.

An attack model is composed of four components, namely a goal function, a transformation, a search method, and a set of constraints (Roth et al. 2021).

Goal Function

Goal function specifies what result the attack model intends to achieve. For classification tasks, the goal functions are usually either targeted or untargeted classification. Targeted classification tries to fool the victim model to classify an example to a specific class different to the correct class, while untargeted classification only aims to make the victim model misclassify.

Transformation

Transformation is how the attack model generates adversarial examples. For token-modification attacks, the transformations are often done by replacing tokens, inserting tokens, deleting tokens, or changing token orders.

One type of transformation is to randomly replace characters in a word or mimic typos (J. Gao et al. 2018; J. Li et al. 2018). These character-level attacks are useful in everyday life applications in a sense that character errors can easily occur. However, they don't actually test the victim model's ability to capture sentiment.

Another common type of transformation is to replace a word with a nearby word in a word embedding space (Jin et al. 2020). The intention of such transformation is to find a synonym replacement. However, in word embedding, synonyms and antonyms are clustered together, so the semantic of the output adversarial example may be opposite to the original sentence. In addition, this method does not take the context and fluency into consideration, resulting in replacements that do not fit in well.

To solve the problem, the latest trend of transformation is to use language models such as BERT (Devlin et al. 2018) and RoBERTa (Liu et al. 2019) to do masked language modelling and generate replacements (Garg and Ramakrishnan 2020; L. Li et al. 2020). The advantage of this approach is that language models take the sentence context into account, so the output adversarial examples will be more natural.

Search Method

Search method is the order of replacing tokens in a sentence, in other words, the sequence of applying transformation to tokens. It could be in document order, in random order, or in importance ranking from most important to least important.

Constraints

Constraints is a set of rules for generated adversarial examples. Part of Speech (POS) constraint requires replacements to have the same POS tag as the replaced tokens to ensure correct grammar. Semantic similarity constraint aims to maintain the semantic of the original sentence. There are also edit distance constraint, performance constraint, to name a few.

1.2 Motivation

The adversarial examples state-of-the-art attack models generate are of low quality. Despite the superior results shown in the papers, after running the attack models ourselves, we observe that most of the adversarial examples the models generate are inconsistent and cause semantic loss. The two main problems are opposite semantic replacements and

Original sentence	no amount of good intentions is able to overcome the triviality of the story	Negative (100%)
Adversarial example	no amount of good intentions is able to overcome the beauty of the story	Positive (99%)

Table 1.1: Examples of attack by BAE on BERT-base classifier.

Original sentence	watching spirited away is like watching an eastern imagination explode	Positive (99%)
Adversarial example	watching spirited away is like watching an eastern magazine explode	Negative (100%)

Table 1.2: Examples of attack by BAE on BERT-base classifier.

irrelevant replacements.

Opposite semantic replacements are antonyms of the original word that alter the semantic of the sentence. Take a look at the examples in Table 1.1 generated by BAE (Garg and Ramakrishnan 2020) attacking victim model BERT-base. Here, ”triviality” is replaced with ”beauty”, which totally changed the semantic of the original sentence from negative to positive. Therefore, even though this is a successful attack to the system, we say this is an invalid attack.

Irrelevant replacements are replacements that have nothing to do with the original word or the sentence context. In this other example in Table 1.2, ”imagination” is replaced with an out-of-context word ”magazine”. The two words are not synonyms, and ”magazine” disrupted the semantic of the original sentence. Such unnatural sentence does not appear in real-life applications, thus this is also viewed as an invalid attack.

1.3 Objective

The goal for this project is to overcome the flaws in previous works and generate high quality adversarial examples. That is to say, we want our attack model to generate examples to be free from opposite semantic or out-of-context replacements and at the same time maintain fluency. Furthermore, we expect our attack model to have higher successful attack rate and lower perturbation than other state-of-the-art attack models.

1.4 Contribution

In this project our contribution is three-folded. First, we figured out that the reason why opposite semantic replacements exist is due to the embedding space of language models, and why out-of-context replacements are so difficult to avoid is because attack models are too general. Unlike previous works which only emphasizes on exterior factors such as adding more constraints, we go further and alter the interior components, which is the language model used in transformation. Secondly, we are the first to generate adversarial examples via a combination of contrastive learning and pretraining. Through pretraining, our attack model is domain-specific, so that instead of generating general replacements, it generates replacements that are related to the sentence context. With the help of contrastive learning, our attack model is capable of separating synonyms and antonyms in the embedding space, which will contribute to generating adversarial examples that are semantically similar to the original sentence. Finally, not only do we have better adversarial examples, but our results also outrun previous works.

Chapter 2

Related Work

2.1 Adversarial Attack for Text

Adversarial attack in NLP has only started to gain its popularity due to the difficulty of generating adversarial text examples caused by the nature of text. As mentioned in the previous section, different attack models have different compositions according to their needs, while they mainly differ in transformation. Our project is inspired by two works: BERT-Attack (L. Li et al. 2020) and BAE (Garg and Ramakrishnan 2020), both of which use language model in transformation.

BERT-Attack aims to generate adversarial examples that are fluent and semantically preserved while having high success rate and minimum perturb percentage. The attack model finds the vulnerable words in a input sentence by masking each word and calculate the output logit. Then, in vulnerability order, use BERT masked language model to generate replacement for each word. BERT-Attack indeed has minimum perturbation. However, when we recreate its experiment, we see a lot of sub-words in its generated adversarial examples. This is because BERT-Attack tokenizes a sentence into sub-word tokens, and some sub-words are not processed correctly.

BAE is the other work we reference to. The objective of BAE is to beat previous baselines on text classification datasets at the same time improve grammatically and semantic coherence via replaceing and/or inserting tokens. BAE uses BERT to predict masked tokens, and apply constraints such as sentence similarity and part of speech tag to ensure fluency. Yet, we still find its adversarial examples unnatural when rerunning its experiment.

2.2 Pretraining

One of the reasons why pretrained language models are so powerful is because they are pretrained on very large corpora from various domains. To illustrate, the pretraining corpus of BERT (Devlin et al. 2018) consists of two huge corpora, BookCorpus and English Wikipedia; RoBERTa (Liu et al. 2019) is pretrained on over 160GB of uncompressed text from books, wikipedia, news articles, web content, and stories.

Recent studies (Gururangan et al. 2020; J. Lee et al. 2020) have shown that a second phase pretraining on domain-specific corpora can largely increase the performance of a language model on tasks in that domain. This is because language models are initially pretrained to perform general tasks, the general corpora may not be diverse enough to include domain specific terms.

After reading these studies, we decided to incorporate this concept and pretrain BERT on domain-specific datasets, hoping it will then generate less out-of-context replacements. The domain we chose to work on is movie reviews.

2.3 Contrastive Learning

One difficulty we face when conducting experiments is that language models cannot distinguish synonyms and antonyms because synonyms and antonyms are clustered together in the embedding space. Therefore, we incorporate contrastive learning in our model. The idea of contrastive learning is to pull similar neighbors together and push away others.

We hope by coupling contrastive learning and BERT, BERT can pull together synonyms, push away antonyms, and generate less opposite semantic replacements.

SimCSE (T. Gao, Yao, and Chen 2021) is a simple contrastive sentence embedding framework we use in this project. Its objective is to improve sentence embedding performance on semantic textual similarity tasks. The paper presents two approaches: unsupervised and supervised. Unspuervised SimCSE predicts an input sentence twice with independently sampled dropout masks to obtain two embeddings as positive pairs. Supervised SimCSE is done by treating entailment pairs from natural language inference datasets as positive instances and add contradiction pairs as hard negatives.

Another contrastive learning framework we reference to is CLINE (Wang et al. 2021). The goal of CLINE is to train a language model that is both defensive against adversarial attacks and sensitive to semantic changes by using both adversarial and contrastive examples, because often times when trying to improve the robustness against adversarial attacks, the performance on contrastive examples fails. CLINE generates positive sentence pairs with the same semantics by replacing extracted words with synonyms, and generate negative sentence pairs with opposite semantics by replacing them with antonyms and random words.

Chapter 3

Methedology

Despite setting multiple constraints and using state-of-the-art language model in transformation in an attack model to ensure the quality of adversarial examples, we can still see a lot of out-of-context and opposite semantic replacements, as demonstrated in previous sections. This is due to the nature of the language models. These contextual word embedding models, such as BERT (Devlin et al. 2018) and ELMo (Peters et al. 2018), are able to create a context-sensitive embedding for each word in a given sentence from pretraining on large text corpora. Unlike traditional word-level vector embeddings like word2vec (Mikolov et al. 2013) and GloVe (Pennington, Socher, and Manning 2014) that treat each word as a independent vector, contextual word embedding models have much more parameters that take the whole sentence context into consideration (Alsentzer et al. 2019). Nevertheless, their embedding spaces are still unable to recognize antonyms since analogies are grouped together. Our proposed method can solve this problem.

We solve the problem from two aspects. For out-of-context replacements, we use pretraining to enhance the language model's ability to generate more domain-specific replacements; for opposite semantic replacements, we adopt the concept of contrastive learning that helps distinguish synonyms and antonyms. We combine contrastive learning and language model pretraining to create our own BERT to use in transformation in our attack model. First

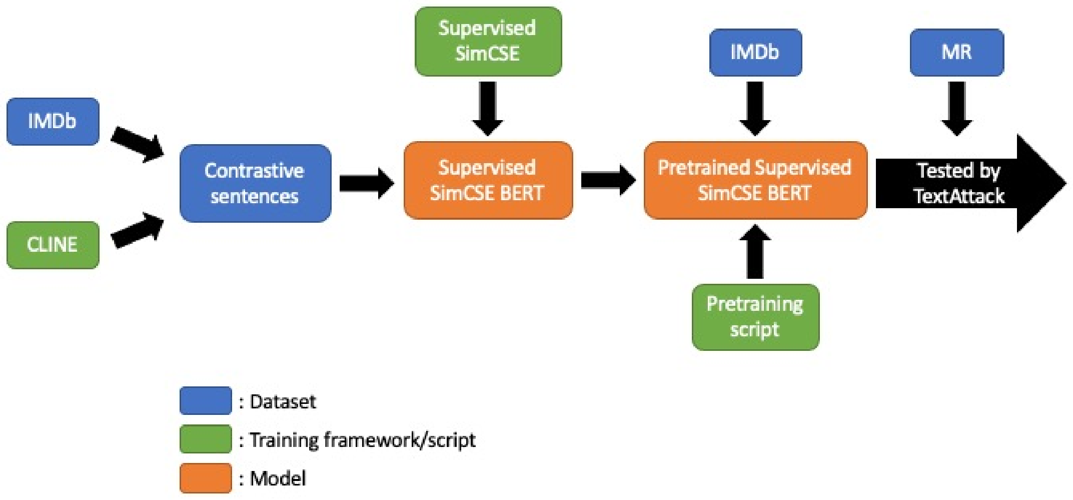


Figure 3.1: Complete flow of creating our own BERT.

we create our own contrastive sentence dataset using the IMDb dataset and CLINE. Next, with that dataset, we train a regular BERT-base using supervised SimCSE. This creates a supervised SimCSE BERT that is capable of separating synonyms and antonyms in the embedding space and avoid generating opposite semantic replacements. Lastly, we pre-train the supervised SimCSE BERT on IMDb to create our final pretrained supervised SimCSE BERT model so that it can generate less out-of-context replacements. Unlike other attack models, our attack model is domain-specific to movie reviews, which is beneficial to generating high quality examples. The complete flow of our method is visualized in Figure 3.1. Datasets involved in the process are colored in blue, training frameworks and scripts we use are colored in green, and the models we create are colored in orange. Arrows indicate what elements are coupled to create another element.

Chapter 4

Experiments

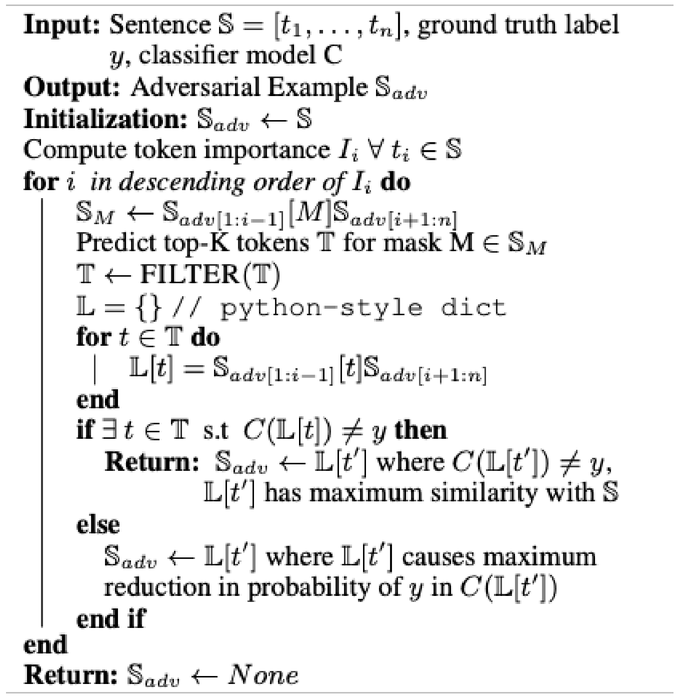
4.1 Datasets

We use two movie review sentiment classification datasets throughout the whole experiment: IMDb (Maas et al. 2011) and MR (Pang and L. Lee 2005). IMDb, also called Large Movie Review Dataset, contains 25,000 highly polar movie reviews for training, 25,000 for testing, and additional 50,000 unlabeled data. MR contains 5,331 positive and 5,331 negative reviews from Rotten Tomatoes. Both of these datasets can be found on Transformers (Wolf et al. 2020).

4.2 Set up

We implement our attack model on BAE (Garg and Ramakrishnan 2020). But instead of using a using a regular BERT in transformation, we use our own BERT in order to see whether modifying BERT is the jey to generating better examples.

We adopt the token replacing algorithm from BAE as shown in Figure 4.1. The goal function is untargeted classification. We replace the regular BERT BAE uses to our own BERT in transformation. Our own BERT varies in different phases of experiment, we will elaborate more in later sections. We use greedy word swap for search method. Greedy



```
Input: Sentence $\mathbb{S} = [t_1, \ldots, t_n]$, ground truth label
       $y$, classifier model C
Output: Adversarial Example $\mathbb{S}_{adv}$
Initialization: $\mathbb{S}_{adv} \leftarrow \mathbb{S}$
Compute token importance $I_i \; \forall \; t_i \in \mathbb{S}$
for $i$ in descending order of $I_i$ do
    $\mathbb{S}_M \leftarrow \mathbb{S}_{adv[1:i-1]}[M]\mathbb{S}_{adv[i+1:n]}$
    Predict top-K tokens $\mathbb{T}$ for mask M $\in \mathbb{S}_M$
    $\mathbb{T} \leftarrow$ FILTER($\mathbb{T}$)
    $\mathbb{L} = \{\}$ // python-style dict
    for $t \in \mathbb{T}$ do
        $\mathbb{L}[t] = \mathbb{S}_{adv[1:i-1]}[t]\mathbb{S}_{adv[i+1:n]}$
    end
    if $\exists \, t \in \mathbb{T}$ s.t $C(\mathbb{L}[t]) \neq y$ then
        Return: $\mathbb{S}_{adv} \leftarrow \mathbb{L}[t']$ where $C(\mathbb{L}[t']) \neq y$,
                $\mathbb{L}[t']$ has maximum similarity with $\mathbb{S}$
    else
        $\mathbb{S}_{adv} \leftarrow \mathbb{L}[t']$ where $\mathbb{L}[t']$ causes maximum
        reduction in probability of $y$ in $C(\mathbb{L}[t'])$
    end if
end
Return: $\mathbb{S}_{adv} \leftarrow None$
```

Figure 4.1: Token replacing algorithm.

word swap measures the importance of each token by masking it and calculate the decrease in probability of predicting the correct label. As for constraints, we include Part of Speech constraint and Universal Sentence Encoder of threshold 0.94 to ensure correct grammar and sentence semantic similarity.

We run attacks with TextAttack (Morris et al. 2020) in all our experiments. TextAttack is a framework designed to enable researchers evaluate different NLP attacks. Given an attack recipe (including a goal function, a transformation, a search method, a set of constraints), a victim model, and a dataset, TextAttack will generate adversarial examples from the dataset using the attack recipe and attack the victim model.

4.3 Results

We modify our BERT as we continue to have new findings. Therefore, in this section, we will separately show the results from different versions of our BERT.

Dataset: MR		
	BAE	Ours
Number of successful attacks	473	**475**
Number of failed attacks	365	**363**
Number of skipped attacks	162	162
Original accuracy	83.8%	83.8%
Accuracy under attack	36.5%	**36.3%**
Attack success rate	56.44%	**56.68%**
Average perturbed word %	13.91%	**13.37%**
Average number of words per input	18.64	18.64
Average number of queries	63.49	**63.19**

Table 4.1: Evaluation of our pretraining only model versus BAE on MR dataset.

4.3.1 Pretraining Only

We start by pretraining a regular BERT-base on IMDb, by running the pretraining script from the official BERT Github repository (https://github.com/google-research/bert) for 50,000 steps, 5,000 warmup steps, and all the other default parameters. We evaluate both BAE and our attack model with TextAttack. The dataset used here is MR and the victim model is BERT-base.

Results are shown in Table 4.1. Overall, our model has a slightly better result. The accuracy under attack is 0.2% lower and the attack success rate is 0.24% higher. The average perturbed word percentage is 0.54% lower, which indicates the replacements are effective, so less perturbations are needed to fool the victim model. Finally, the average number of queries is 0.3 lower, implying our attack model needs less transformation to each token to generate an adversarial examples.

After studying the adversarial examples from both attack models, some presented in Table 4.2, we find out that pretraining BERT does have an effect on generating token replacements. The replacements generated by our model are more related to movies. However, the problems we are trying to solve has not improved. There are still a considerable amount of opposite semantic and out-of-context replacements in the adversarial examples

Original sentence	the movie is a little tired; maybe the original inspiration has run its course	Negative (100%)
BAE	the mind is a little tired; yet the original memory has continued its course	Positive (100%)
Ours	the beginning is a little tired; maybe the original tale has improved its course	Positive (88%)
Original sentence	one of the funnier movie in town	Positive (94%)
BAE	one of the funnier locations in town	Negative (97%)
Ours	one of the funnier scenes in town	Negative (99%)

Table 4.2: Examples of attacks by our pretraining only model versus BAE on BERT-base classifier.

Dataset: IMDb		
	BAE	Ours
Number of successful attacks	**583**	365
Number of failed attacks	**338**	556
Number of skipped attacks	79	79
Original accuracy	92.1%	92.1%
Accuracy under attack	**33.8%**	55.6%
Attack success rate	**63.3%**	39.63%
Average perturbed word %	4.06%	**3.43%**
Average number of words per input	233.5	233.5
Average number of queries	425.07	**373.2**

Table 4.3: Evaluation of our pretraining only model versus BAE on IMDb dataset.

from our model. The slightly better result is not sufficient for proving our model is superior.

We also evaluate BAE and our model using IMDb dataset, results shown in Table 4.3. In contrast to using MR, here our model has a much lower attack success rate than BAE. The reason is because our BERT is pretrained on IMDb, so the replacements it generates are too similar to the original tokens that they are unable to fool the victim model.

4.3.2 Contrastive Learning and Pretraining

Next, we add contrastive learning to our attack model. Instead of pretraining a regular BERT-base, now we pretrain supervised SimCSE BERT-base (T. Gao, Yao, and Chen

Dataset: MR					
	BAE	Ours (50,000)	Ours (25,000)	Ours (5,000)	Ours (2,500)
Number of successful attacks	473	471	473	487	**501**
Number of failed attacks	365	367	365	351	**337**
Number of skipped attacks	162	162	162	162	162
Original accuracy	83.8%	83.8%	83.8%	83.8%	83.8%
Accuracy under attack	36.5%	36.7%	36.5%	35.1%	**33.7%**
Attack success rate	56.44%	56.21%	56.44%	58.11%	**59.79%**
Average perturbed word %	13.91%	13.19%	**13.13%**	13.58%	13.17%
Average number of words per input	18.64	18.64	18.64	18.64	18.64
Average number of queries	63.49	64.27	64.05	64.01	**62.96**

Table 4.4: Evaluation of our contrastive learning and pretraining model of different number of pretraining steps versus BAE on MR dataset.

2021) on IMDb by running the same pretraining script as before for 50,000 steps, 5,000 warmup steps and all the other default parameters.

We discover that the result has not improved a lot. In fact, after analyzing the adversarial examples, we find out that there are still a lot of opposite semantic replacements. We suspect that this is due to excessive pretraining. SimCSE is pretrained too much that its contrastive learning characteristics are hiddened. Therefore, we lower the number of training steps and warmup steps.

From Table 4.4 we can see, the lower the number of training steps, the better the result. The one trained with 2,500 steps has a overall better performance, with 2.8% lower accuracy under attack, 3.35% higher attack success rate, and 0.53 lower average number of queries than BAE. This is also reflected in the generated adversarial examples, as shown in Table 4.5. Although opposite semantic and out-of-context replacements are not completely eliminated, we see less such replacements, and a much higher attack success rate.

Original sentence	fans of the modern day hong kong action film finally have the worthy successor to a better tomorrow and the killer which they have been patiently waiting for	Positive (100%)
BAE	fans of the modern day hong kong action film finally have the only successor to a better tomorrow and the killer which they have been helplessly waiting for	Negative (99%)
Ours (50,000)	fans of the modern day hong kong action film finally have the disappointing successor to a better tomorrow and the killer which they have been patiently waiting for	Negative (51%)
Ours (25,000)		Failed
Ours (5,000)		Failed
Ours (2,500)	fans of the modern day hong kong action movie now have the usual successor to a better tomorrow and the killer which they have been already waiting for	Negative (83%)

Table 4.5: Examples of attacks by our contrastive learning and pretraining model of different number of pretraining steps versus BAE on BERT-base classifier.

4.3.3 Using CLINE to Create Contrastive Sentences

Now that we have proved our goal can be achieved by contrastive learning and pretraining, the next thing we will do is to create contrastive sentences using CLINE (Wang et al. 2021). CLINE generates a semantically close sentence by replacing words in the original sentence with synonyms, hypernyms and morphological changes, and generates a semantically opposite sentence by replacing words with antonyms and random words. We are not using SimCSE to create contrastive sentences because its algorithm is questionable.

To start with, we run the word replace script from the official CLINE Github repository (https://github.com/kandorm/CLINE) with the IMDb dataset to generate contrastive sentences. Then, using the contrastive sentences, we run supervised SimCSE training script from the official SimCSE Github repository (https://github.com/princeton-nlp/SimCSE) to train a supervised SimCSE BERT. Finally, we pretrain the model for 2,500 steps like we did before on IMDb to get the final pretrained supervised SimCSE BERT.

Dataset: MR			
	BAE	Ours (pre-training only)	Ours (IMDb contrastive sentences)
Number of successful attacks	473	475	**495**
Number of failed attacks	365	363	**343**
Number of skipped attacks	162	162	162
Original accuracy	83.8%	83.8%	83.8%
Accuracy under attack	36.5%	36.3%	**34.3%**
Attack success rate	56.44%	56.68%	**59.07%**
Average perturbed word %	13.91%	**13.37%**	13.5%
Average number of words per input	18.64	18.64	18.64
Average number of queries	63.49	**63.19**	63.77

Table 4.6

One problem we faced during the process now is that the word replace script from CLINE runs extremely slowly. This is because the IMDb dataset has on average 233.5 word per input, and the replace ratio in the script is set to 0.5, which means at most half of a sentence will be replaced. In addition, we find out that random word replacement takes too much time to process and will introduce uncertainties. To fix this efficiency problem, we change the replace ratio to 0.05 and remove random word replacement for semantically opposite sentences. From here we get 26,370 pairs of sentences.

From Table 4.6 we can see the result outperforms the baseline, but it is not good enough. We suspect this is due to insufficient training contrastive sentences. It is also possible that the training strategy SimCSE uses is not suitable for our goal.

Chapter 5

Conclusion and Future Work

In this project, we propose a new method that combines contrastive learning and pretraining to generate high quality adversarial examples. This is an ongoing project. In this term, we discovered that pretraining and contrastive learning have positive effects on generating high quality adversarial examples. We started by using a second-phase pretraining to make our attack model domain-specific, which has shown to be effective. Then we went one step further and added in contrastive learning to alter the embedding space, which improves the results even more. For the next term, we plan to continue the development of our model. We want to find a better way to combine contrastive learning and pretraining. We will run a larger scale experiment, and if possible, involve human evaluation to demonstrate the effectiveness of our work.

Bibliography

Szegedy, Christian et al. (2013). “Intriguing properties of neural networks”. In: *arXiv preprint arXiv:1312.6199*.

Goodfellow, Ian J, Jonathon Shlens, and Christian Szegedy (2014). “Explaining and harnessing adversarial examples”. In: *arXiv preprint arXiv:1412.6572*.

Roth, Tom et al. (2021). “Token-modification adversarial attacks for natural language processing: A survey”. In: *arXiv preprint arXiv:2103.00676*.

Gao, Ji et al. (2018). “Black-box generation of adversarial text sequences to evade deep learning classifiers”. In: *2018 IEEE Security and Privacy Workshops (SPW)*. IEEE, pp. 50–56.

Li, Jinfeng et al. (2018). “Textbugger: Generating adversarial text against real-world applications”. In: *arXiv preprint arXiv:1812.05271*.

Jin, Di et al. (2020). “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*. Vol. 34. 05, pp. 8018–8025.

Devlin, Jacob et al. (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805*.

Liu, Yinhan et al. (2019). “Roberta: A robustly optimized bert pretraining approach”. In: *arXiv preprint arXiv:1907.11692*.

Garg, Siddhant and Goutham Ramakrishnan (2020). “Bae: Bert-based adversarial examples for text classification”. In: *arXiv preprint arXiv:2004.01970*.

Li, Linyang et al. (2020). "Bert-attack: Adversarial attack against bert using bert". In: *arXiv preprint arXiv:2004.09984*.

Gururangan, Suchin et al. (2020). "Don't stop pretraining: adapt language models to domains and tasks". In: *arXiv preprint arXiv:2004.10964*.

Lee, Jinhyuk et al. (2020). "BioBERT: a pre-trained biomedical language representation model for biomedical text mining". In: *Bioinformatics* 36.4, pp. 1234–1240.

Gao, Tianyu, Xingcheng Yao, and Danqi Chen (2021). "SimCSE: Simple Contrastive Learning of Sentence Embeddings". In: *arXiv preprint arXiv:2104.08821*.

Wang, Dong et al. (2021). "Cline: Contrastive learning with semantic negative examples for natural language understanding". In: *arXiv preprint arXiv:2107.00440*.

Peters, Matthew E. et al. (2018). *Deep contextualized word representations.* arXiv: `1802.05365 [cs.CL]`.

Mikolov, Tomas et al. (2013). *Efficient Estimation of Word Representations in Vector Space.* arXiv: `1301.3781 [cs.CL]`.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Alsentzer, Emily et al. (2019). "Publicly available clinical BERT embeddings". In: *arXiv preprint arXiv:1904.03323*.

Maas, Andrew L. et al. (June 2011). "Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.* Portland, Oregon, USA: Association for Computational Linguistics, pp. 142–150. URL: `http://www.aclweb.org/anthology/P11-1015`.

Pang, Bo and Lillian Lee (2005). "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales". In: *Proceedings of the ACL.*

Wolf, Thomas et al. (Oct. 2020). "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural*

Language Processing: System Demonstrations. Online: Association for Computational Linguistics, pp. 38–45. URL: `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

Morris, John X et al. (2020). "Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp". In: *arXiv preprint arXiv:2005.05909*.