Adversarial Examples for Text Classification

Anonymous Author(s)
Affiliation
Address
email

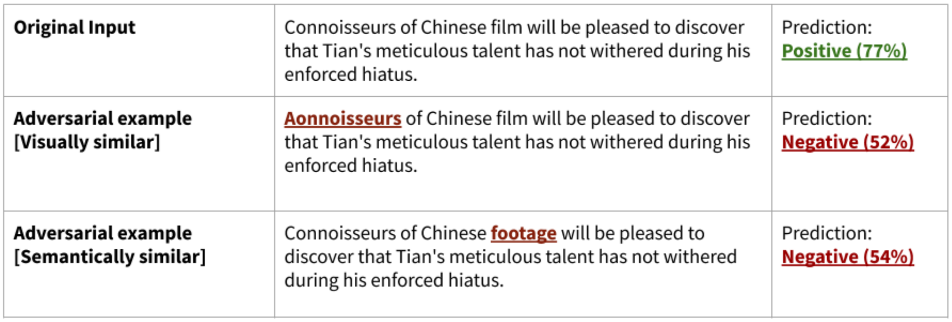
Abstract

Natural Language processing helps the computer to understand the way that the humans write and speak as it is a very complex task because of the involvement of large amount of unstructured data. Through Natural Language Processing, computer can effectively communicate with the humans in their own language, along with it also helps to scale the language related tasks as well. As Natural Language Processing makes it easy for the computer to read the text, hear the speech, measure the sentiment and also determines which parts are important. Now a days, with the evolution of Natural Language Processing, machines can now easily process large number of language based data than the humans in a more compatible and objective way. As a large amount of un structured data is being generated each day, from different social media sites, websites, through different contents, from office records to the medical records, so there is an need of an automation that can fully analyze the text data in an efficient way. The major motivation behind working over Natural Language Processing is that it makes it possible for the computers to read the text, hear the speech, measures the sentiment and also determines which part are important as well. Modern text classification models are basically vulnerable to the opposite example. Discomposed versions of the original text which are indistinguishable by the humans get misclassified by the models. In the recent research, rule based synonym replacement strategies have been considered to generate the adversarial examples. But the limitation of this approach is that it leads to out of the context and very complex tokens replacements. In this research, we formulate attacks against a trained model (LSTM Neural Networks, Naive Bayes Classifier, Random Forest Classifier) by testing it against inputs that are semantically similar to the original but have slight paraphrasing or synonym substitutions. Through result analysis, it can be seen that the model trained on the augmented data performed better and gives better accuracy, against perturbations and yielded nearly double the amount of failed attacks.

1 Introduction and Motivation

Before discussing adversarial text attacks in NLP we need to know what Adversarial Examples are, adversarial example is a fake data that mimics the training data but produces misclassified label when the machine learning algorithm encounters it. On the other hand adversarial attack is an pipeline which generates the adversarial examples Moustafa Alzantot et al. [1].

Deep Neural Network has been very popular since from the start and AlexNet performance increased the hope in deep learning. One of the major problems faced in deep neural network is to design such



Original Input	Connoisseurs of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Positive (77%)**
Adversarial example [Visually similar]	**Aonnoisseurs** of Chinese film will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (52%)**
Adversarial example [Semantically similar]	Connoisseurs of Chinese **footage** will be pleased to discover that Tian's meticulous talent has not withered during his enforced hiatus.	Prediction: **Negative (54%)**

Figure 1: Adversarial Examples in NLP

an deep neural network that is robust at detecting the changing and resisting against the adversarial examples Jacob Devlin et al. [2].

The concept of the black box attack was originally introduced in the computer vision domain which is easier to mimic the original data by infusing fuzzy data (noise) to the original data represented in the continuous pixel intensity. Unlike, the computer vision, in NLP we have an concept of tokens, NLP utilizes the discrete word as tokens and it is often very challenging to swap out tokens without changing the meaning of the sentences, sentiments, implications and syntax of a sentence Javid Ebrahimi et al. [3].

The motivation behind working on the adversarial text attacks in NLP is that, usually the data is imbalanced and to balance the data usually Up Sampling and Text Augmentation, SMOTE, ENN and KNN are performed to balance the data. In some of these approaches the data is balanced by the generating the copy of the data from the original data. In some cases the data is balanced by inserting, deleting words, replacing them with synonyms and so. But it must be noted that swapping an input's words or randomly deleting several letters can severely alter the performance of our Natural Language Processing models (NLP). We formulate attacks against a trained model (i.e LSTM Neural Networks, Random Forest Classifier, Naive Bayes Classifier)by testing it against inputs that are semantically similar to the original but have slight paraphrasing or synonym substitutions. In this research Text Attack library is used and an adversarial attack is run on each of the trained models (the one trained with up sampled data and the one with augmented data). The attack will run until 1000 attacks are successful at fooling each model.

The research contribution is as follows

• By using the text attack library an adversarial attack is run on each of the trained models i.e (LSTM Neural Network, Random Forest Classifier, Naive Bayes Classifier), considering both the trained models, i.e. the one trained with up- sampled data and the one trained with the Augmented Data.

• A performance comparison of each of the model is done considering both the scenarios, i.e. the model trained with the up-sampled data and the model trained with the Augmented data. Along with this, while considering the accuracy of each of the model. The original accuracy, accuracy under attack, the number of successful attacks, the number of failed attacks, the number of skipped attacks are considered as well

2 Related Work

The recent research have shown the possibility of the machine learning models being attacked or exposed to adversarial attacks i.e. it is an approach which tries to deceive/ fool the model with the deceptive data and it has emerged as an major challenge in the field of artificial intelligence and machine learning, small input perturbation in the system, i.e. small change in the system which may be as a result of third object interacting with the system, and it results in the misclassification by the machine learning model. Adversarial attacks or adversarial example is an major challenge in the field of Natural Language processing and it is quite a complex and major problem as compared to computer vision tasks.

In the initial work, the text models were attacked considering the introduction of the error at the character level or adding or deleting the words. These approaches result in an very un natural adversarial examples that lacks the grammatical correctness and thus can be easily identified by the humans.

The rule based synonym replacement approaches have resulted in generating more natural looking adversarial examples. Nicolas Papernot et al. combined all the previous approaches and proposed an TextFooler which is an strong black box attack baseline used in the text classification model. The adversarial examples being generated through the Textfooler only considers the token level similarity using the word embeddings and does not considers the overall sentence structure, which results in a out of context and an very un naturally complex replacement.

In extension of topics in biomedical decision making, vulnerabilities in biomedical NLP could be devastating for medical decision tasks. If wrong decision is proposed by AI, it will negatively impact the patients' health. Adversarial Examples for Biomedical NLP Tasks discusses adversarial examples generated by BERT-based model in BioNLP

Besides performing tasks in text classification in general NLP domains,it is also important to leverage existing techniques for improving detections of adversarial examples in other domains, ie. medical domain in support of fraudulent billing activities.

In this paper we have proposed a novel approach i.e. BERT based Adversarial example, which is an adversarial example generation technique considering the BERT masked language model that replaces the words in a better way which fits the overall context of the English language. Along with the replacing the words, we also add new tokens in the sentence to improve the attacking strength of BAE. These perturbations in the input sentence are being achieved by masking some part of the input and using the LM to fill up the mask. In the proposed approach text attack library is used and an adversarial attack is run on each of the trained models (the one trained with up sampled data and the one with augmented data). The attack will run until 1000 attacks are successful at fooling each model.

3 Datasets and Methods

To analyze the performance of the proposed approach the dataset considered for the implementation is Women's E-Commerce Clothing Reviews. The data-set is available publically on Kaggle. The dataset contains 23486 rows and 10 feature variables. In the proposed approach only two feature variables are considered i.e "Review Text" and "Recommended IND", as this paper is focused on text classification.

In the "Recommended IND" column, the 1 represents the "Recommended" and 0 represents the "Not Recommended". If we take a closer look at the "Recommended IND" column, we can see that the data is imbalanced, to solve this issue, Up Sampling and Text Augmentation is performed to balanced the dataset.

Four different machine learning algorithms are used to train the model on the given dataset. The model used for the training include the LSTM Neural Network, Random Forest Classifier and Naive Bayes Classifier. Each of the model is trained considering bot the up sampled data as well as the augmneted data.

In the next step, using the TextAttack library, an adversarial attack is run on each of the trained models (the one trained with upsampled data and the one with augmented data). The attack will run until 1000 attacks are successful at fooling each model.

In the proposed approach, four d

on multiple text classification tasks considering the Amazon and IMDB datasets, which are famous sentiment/ text classification datasets. The text classification models used in the implementation are word-LSTM, word-CNN and a fine tuned BERT. Each of the above mentioned models are trained considering the training data and the performance of each of the models are evaluated considering the test data on which the adversarial attacks are performed

To generate adversarial examples we define two types of perturbations on the input, One is to replace the token with the another, and other perturbation is to insert a new token while preserving the other words in the sentence. Four types of different attack modes can be constructed by the combination of these two perturbation types, the first one is only replacement (BAE-R), the second one is only insertion, third one is replace or insertion (BAE-R/I) and the fourth one is both replace and insertion (BAE- R + I). All the four attack modes are listed in the Figure 2.

We present 4 attack modes for BAE based on the R and I operations, where for each token t in S:

- BAE-R: Replace token t

- BAE-I: Insert a token to the left or right of t

- BAE-R/I: Either replace token t or insert a token to the left or right of t

- BAE-R+I: First replace token t, then insert a token to the left or right of t

The Replace (R) and Insert (I) operations are performed on a token t by masking it and inserting a mask token adjacent to it respectively. The pretrained BERT-MLM is used to predict the mask tokens as shown in Figure 1. BERT-MLM is a powerful LM trained on a large training corpus (2 billion words), and hence the predicted mask tokens fit well into the grammar and context of the text.

If we pass an sentence s with the ground truth y and classifier c, the BERT based adversarial example algorithm will generate an adversarial example Sadv as an output. While evaluating the tokens in the sentences, this algorithms aims to generate an adversarial example by perturbing around the highest important tokens to increase the efficiency. There are multiple approaches to do the importance ranking which includes genetic algorithms, detection, replacement. In this research we have used masking. The BAE-R pseudo code is explained in Figure. 3.

Essentially, important tokens have higher influence on prediction tasks than the less important one (ie. Determiner ‘The’). By masking, we could replacing the masked word in S with an arbitrary word and see how it affects the prediction. The greater the change, the greater the token importance. There are another masking step, but it is different than the one used in importance ranking.

Going back to the algorithm, the token importance returns the token index ranked in descending order. In this loop, mask sentence is generated on a token. Then, the BERT model will predict a set of top-K tokens, T, for the masked word. Unlike the previous mask, this mask step is used as a target for finding similar tokens as captured in the BERT embeddings space. Noted BERT embedding isnt perfect also.

Using [t], it generates a set of new sentences L by original unmask tokens with [t] at mask’s position. By feeding the new sentences L into the classifier, we can compute the classification C(L(t)), determine whether it is a successful attack by comparing with y, and return if found. If no classification is found at that run, best unsuccessful adversarial example will carry over to next round for another perturbation. Eventually, it will loop through all combinations to find mis classified cases if it exists.

Now that we have adversarial examples generated, in this research we targeted different models, Word-LSTM, Word-CNN, and BERT considering multiple datasets as illustrated in Figure 4. The performance of the attacks are evaluated on performing text classification tasks including sentiment classification, subjectivity detection, and question type classifications; and the quantitative metrics used are the percentages of accuracy and maximal perturbations, grammatical correctness, and sentimental accuracy.

The BAE-attacks are compared to the TEXTFOOLER. Long story in short, TEXTFOOLER shares great similarity to the approach from the current paper except it uses a fixed vector embedding of synonyms(50), and fixed threshold of word similarity greater than 0.80. The ground truth label is evaluated by humans using grammatical correctness and sentiment analysis on the sentence using 2 human evaluators. The BERT-MLM, however, does not guarantee semantic coherence to the original text as demonstrated by the following simple example. Consider the sentence: 'the food was good'. For replacing the token 'good', BERT-MLM may predict the token 'bad', which fits well into the grammar and context of the sentence, but changes the original sentiment of the sentence. To achieve a high semantic similarity with the original text on introducing perturbations, we filter the set of top K tokens (K is a pre-defined constant) predicted by BERT-MLM for the masked token, using a Universal Sentence Encoder (USE) based sentence similarity scorer. For the R operation, we additionally filter out predicted tokens that do not form the same part of speech (POS) as the original token.

4 Experiment and Results

The first step, involves installing all the required libraries and importing them. After this the data-set is loaded and the data analysis and visualization is performed to find the hidden insights in the data and to analyze the data. The data-set contains 23486 rows and 10 feature variables. In the proposed approach only two feature variables are considered i.e "Review Text" and "Recommended IND", as this paper is focused on text classification. The "1" in the "Recommended IND" represents the recommended and the "0" represents the non recommended. After the data analysis and visualization the data-set is cleaned, all the missing values are removed and outliers are handled as well. The "Recommended IND" column contains the imbalanced data i.e the number of 1's which represents the recommended are more than the 0's which represents the non recommended. In this case if we do not balance the data-set. The probability of prediction accuracy of "0" will be very less as compared to the "1". So, it is needed to balance the dataset. There exists multiple approaches to balance the dataset which included ENN< KNN, Up Sampling and Text Augmentation technique. Each technique has its own pros and cons. In this research, Up sampling and text augmentation are considered to increase the number of 0 in the "Recommended IND" column from 4101 to 16104. Up sampling technique works by generating the copies of the original data and increasing the data size. While in the text augmentation, words are randomly swapped, deleted, as well as replaced or inserted with synonyms using pretrained word embeddings. In this research Easy Data Augmentation technique is used.

After this LSTM Neural Network, Naive Bayes Classifier and Random Forest Classifier model is trained on this dataset (with upsampled data and the one with augmented data). The evaluation of all three models is done considering the accuracy score, percision score, recall score and the f1-score.

The data set is spllited into the train and test set, and 80 percent of the data is used for the training purpose. After converting he text into the vectors and training the LSTM neural network on the training data set and evaluating the model on the test data 85 percent is obtained on the test, along with an precision of 80 percent for the 0 and 89 percent for the 1. In this case no adversarial attack is run on the trained model. The confusion matrix obtained is as follows as shown in Figure 2.

In the next step, after converting the text into vectors using the TF-IDF and training the model using Random Forest Classifier. From the results it can be seen that an accuracy of 97 percent is obtained on the test data set. Along with this a precision of 0.94 is obtained for the 0 and a precision of 0.99 for the one. Following is the confusion matrix obtained as shown in the Figure 3

Bag of Words can also be used to convert text into vectors, after converting the text into vectors using count vectorizer and implementing the Naive Bayes Classifier on the training dataset and evaluating the model on the test data set. Following is the confusion matrix obtained from the Naive Bayes Classifier model on the test data set as shown in Figure 4.

In the next step, using the TextAttack library, an adversarial attack is run on each of the trained models i.e. LSTM Neural Network, Random Forest Classifier and the Naive Bayes Classifier (the one

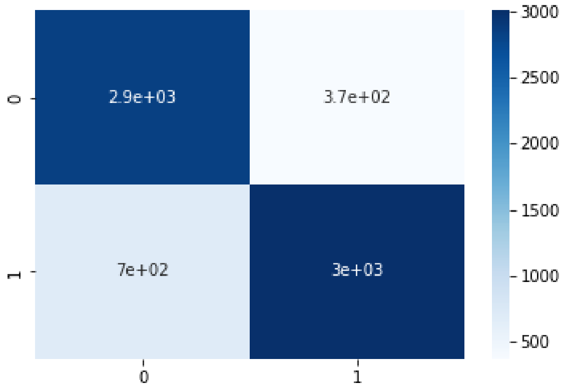


Figure 2: Confusion Matrix for the LSTM Neural Network with no adversarial attack runned

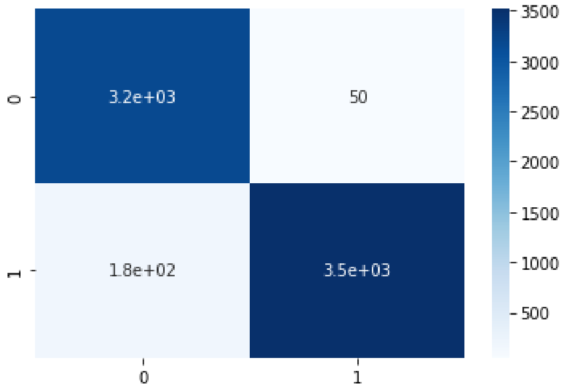


Figure 3: Confusion Matrix for the Random Forest Classifier

trained with upsampled data and the one with augmented data). The attack will run until 50 attacks are successful at fooling each model and is shown in Figure 5 as well.

From the results it can be analyzed that the model trained on a dataset with augmented data outperformed all-around. In comparison, it had significantly better accuracy against perturbations and yielded nearly double the amount of failed attacks.

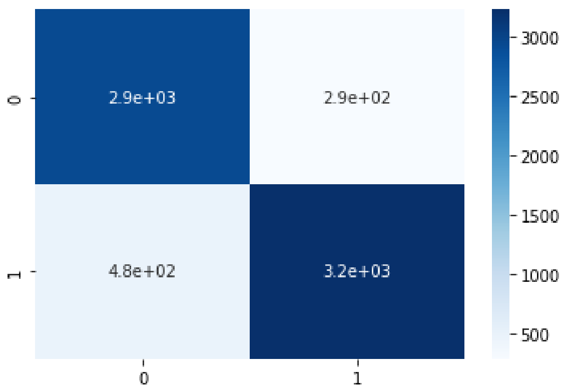
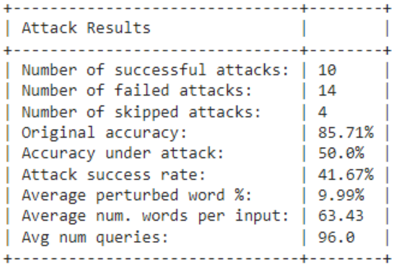
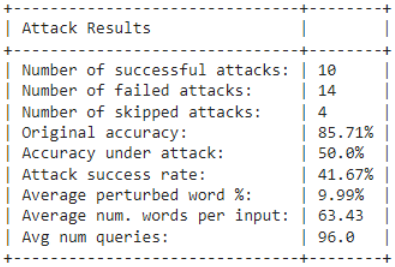


Figure 4: Confusion Matrix for the Naive Bayes Classifier



```
+-------------------------------+--------+
| Attack Results                |        |
+-------------------------------+--------+
| Number of successful attacks: | 10     |
| Number of failed attacks:     | 14     |
| Number of skipped attacks:    | 4      |
| Original accuracy:            | 85.71% |
| Accuracy under attack:        | 50.0%  |
| Attack success rate:          | 41.67% |
| Average perturbed word %:     | 9.99%  |
| Average num. words per input: | 63.43  |
| Avg num queries:              | 96.0   |
+-------------------------------+--------+
```

Figure 5: Attack Results using the LSTM Neural Network considering the Up Sampled Data set



```
+-------------------------------+--------+
| Attack Results                |        |
+-------------------------------+--------+
| Number of successful attacks: | 10     |
| Number of failed attacks:     | 14     |
| Number of skipped attacks:    | 4      |
| Original accuracy:            | 85.71% |
| Accuracy under attack:        | 50.0%  |
| Attack success rate:          | 41.67% |
| Average perturbed word %:     | 9.99%  |
| Average num. words per input: | 63.43  |
| Avg num queries:              | 96.0   |
+-------------------------------+--------+
```

Figure 6: Attack Results using the LSTM Neural Network considering the Augmented Text Data set

Discussion and Conclusion

In this research, adversarial text attacks were formulated against a trained model (LSTM Neural Networks, Naive Bayes Classifier, Random Forest Classifier) by testing it against inputs that are semantically similar to the original but have slight paraphrasing or synonym substitutions. Through result analysis, it can be seen that the model trained on the augmented data performed better and gives better accuracy, against perturbations and yielded nearly double the amount of failed attacks.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section 3.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? **[TODO]**

    (b) Did you describe the limitations of your work? **[TODO]**
    (c) Did you discuss any potential negative societal impacts of your work? **[TODO]**
    (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[TODO]**

2. If you are including theoretical results...
    (a) Did you state the full set of assumptions of all theoretical results? **[TODO]**
    (b) Did you include complete proofs of all theoretical results? **[TODO]**

3. If you ran experiments...
    (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]**
    (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? **[TODO]**
    (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[TODO]**
    (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[TODO]**

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
    (a) If your work uses existing assets, did you cite the creators? **[TODO]**
    (b) Did you mention the license of the assets? **[TODO]**
    (c) Did you include any new assets either in the supplemental material or as a URL? **[TODO]**
    (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[TODO]**
    (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[TODO]**

5. If you used crowdsourcing or conducted research with human subjects...
    (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[TODO]**
    (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[TODO]**
    (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[TODO]**

References

[1] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics. (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[3] Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, pages 31–36, Melbourne, Australia. Association for Computational Linguistics. **15**(7):5249-5262.

[4]Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In Proceedings of ACL.

[5]Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical black-box attacks against machine learning. In Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ASIA CCS '17, pages 506–519, New York, NY, USA. ACM.

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.