Generating Part-of-speech Focused Adversarial Examples for Sentiment Analysis

Author:
Kathleen de Boer
6439608

Supervisor:
Dr. M.P. Schraagen

Second Supervisor:
Prof. dr. G.K. Keller

A 7.5EC thesis for the
Bachelor of Science in Artificial Intelligence

Faculty of Humanities
Utrecht University
The Netherlands
July 2, 2021

Generating Part-of-speech Focused Adversarial Examples for Sentiment Analysis

Abstract

The focus of the contents presented in this paper is on the generation of so-called *black-box* adversarial examples for the natural language processing (NLP) task *sentiment analysis*. The adversarial examples are created through synonym-substitution of words with certain *Part Of Speech* (POS) tags. The effectiveness of a variety of POS tags is tested, while preserving the semantical similarity of the modified text. Naive Bayes (NB) models and convolutional neural networks (CNN) are two popular models commonly used in sentiment analysis. These models are trained and tested on the IMDB movie review dataset. The effectiveness of the proposed method and the resilience of the models is evaluated by the measure of decrease in accuracy. The results of the experiment show that the selection of the Part-of-speech tags *Adjective*, *Adverb*, and *Modal* generates successful and effective adversarial examples with little distortion to the original text. Additionally, the proposed method uncovers vulnerabilities in both tested models.

Keywords: Sentiment Analysis, Adversarial Examples, Part-of-speech Tags, Convolutional Neural Network, Naive Bayes

Contents

Chapter 1

Introduction

With the rise of the internet over the last few decades came widespread adoption of social media, news sites, online forums and product review websites. These platforms make it possible for anyone to let their opinions about any subject, whether it is a product, service or company, be heard by the mass. The sheer volume of this data leads to data overload. Morover, such data is not useful owing to its unstructured nature. For example, were a company to analyse all feedback on a product by hand, this would lead to an enormous amount of work. Sentiment analysis can automate and aid in this process of analyzing unstructured textual data. Another important application of Sentiment analysis is the monitoring of social media. If some brand would one day see a spike in mentions on Twitter, for example, this does not immediately imply that their popularity has grown. Not all attention is positive attention and the mentions could be a sign of a negative opinion held by masses.

Undoubtedly, sentiment analysis is of major value and relevance in modern society and this is unlikely to go away. Sentiment analysis, in its current form is a combination of two important sub fields within artificial intelligence: machine learning and natural language processing. Machine learning models used for sentiment analysis tend to perform well in practice. However, given their relevance in society, it is important to find weaknesses in such model in order to improve upon their robustness. The widely applied transformer architecture and state-of-the-art model BERT tends to be vulnerable to imperfect and noisy data [34], caused by typewriting mistakes for example. Although such mistakes may happen accidentally, certain similar modifications to input-texts can be created with malicious intent. These so-called adversarial attacks can be developed to exploit vulnerabilities in models. Additional research is thus required to find such weaknesses prematurely, in order to prevent adversarial attack by better understanding them. Moreover, newly generated examples can be used to augment training datasets. Training a model using adversarial examples has been shown to benefit robustness and generalization of the model. [24].

In this study the performance of two popular machine learning models will be analysed with the help of adversarial examples. Adversarial examples are slight modifications of correct data examples from the same data distribution, that will make the machine learning model incorrectly classify them, although human would still be able to give the correct classification. In the context of sentiment analysis and natural language processing, this would mean that upon changing a sentence, a model would assign a different sentiment or translation although the new sentence is semantically similar. If a ma-

chine learning model fails to perform well on the modified examples, this is indicative of vulnerability to adversarial examples.

Adversarial attacks have a long history of study and application mostly in image recognition, for which they can be generated very successfully. For natural language processing tasks such as sentiment analysis however, little prior research exists. Adversarial examples for natural language processing (NLP) are different than those for image recognition as different constraints have to be taken into account. The modifications on a text are successful if they are sufficiently similar to the original text. This is why, in this study, semantic similarity of the adversarial examples is stressed. Furthermore, for the sake of readability and similarity, it is important that only a small selection of words is modified. In this study, a method for generating adversarial examples is proposed in which certain part-of-speech tags will be replaced by their synonyms. Previous research shows that adverbs, adjectives and modals contribute the most to a sentence sentiment [37] [14] [4]. The aim of the current study is to reveal possible vulnerabilities in Naive Bayes and convolutional neural network models using the proposed method.

Chapter 2

Theoretical background and Related work

This chapter contains a review on related work and a theoretical background on adversarial examples, sentiment analysis and part-of-speech tagging. Some influential studies on the subjects will be discussed.

2.1 Adversarial examples

Recent research has shown that deep neural networks and other machine learning models can be fooled by adversarial examples [12]. Recall that adversarial examples are specific inputs that are made to fool models. This technique can be used to deceive models such as those used in spam-filtering systems [20] or a self-driving cars [35], with possibly grave consequences. For example, self driving cars use image recognition. A possible attack could be the specific editing of a stop sign so that the systems in the car will perceive the stop sign as something else. In the case of the spam-filtering task, knowledge of the spam filter could make a spam classifier misclassify slightly modified e-mails as non-spam with the consequence that spam or dangerous mail could end up in in the inbox. As stated, adversarial attacks have already been proven to be an effective method in image recognition. The effectiveness of this technique and the lack of a dependable defence, as almost all defenses are shown to be effective only for part of attacks, show the vulnerabilities of machine learning models [39].

Adversarial attacks can be either targeted or un-targeted. Targeted means that the attacker tries to have the model predict a specific desired class. Un-targeted attacks are certain attacks where the attacker tries to deceive the model to predict any of the incorrect classes. Additionally, attacks can be categorized into two categories; black-box and white-box attacks. A black-box attack has no information on the models parameters, architecture or training data. These attacks are made without knowledge of the target model by modifying the input and observing the prediction of the classifier. Black-box attacks are possible because of a certain property of some adversarial examples, called transferability. Adversarial examples with transferability can fool multiple different machine learning models, or the same model but trained on different datasets. Research from Liu et al. shows that un-targeted, black-box attacks transfer easily, but targeted adversarial examples almost never transfer

with their target labels. In white-box attacks the attacker may have various degrees of knowledge about the target model, such as the important features used in classification. Commonly used white-box attacks for image classification make use of the target classifier's gradient [12] [21]. Gong et al. proposed to incorporate this technique to the field of text classification. They searched for adversaries in the embedding space, and then reconstructed the adversarial texts via nearest neighbor search. Words that are close in the embedding space however, are not automatically similar semantically.

When constructing adversarial examples for text classification, small perturbations in a text can make it unreadable. Input text often becomes grammatically incorrect and semantically wrong. The method of using word-embedding by Gong et al. often produced improperly constructed sentences. This is why different methods have to be developed for the generation of textual adversarial examples.

2.1.1 Adversarial Attacks for Text Classification

Different approaches to generating adversarial examples for text classification have been attempted. The approaches mentioned in this section all try to preserve semantically and syntactically correct adversaries. There is no widespread consensus on a measure of sufficient similarity. Modifications can be made either on the character level of a sentence, which results in visual similarity, or on the level of words. Goodman et al., following Gao et al., proposed FastWordBug, a framework that can generate black-box adversarial text using small perturbations that have a typo-like appearance as can be seen in Figure 2.1. Although this method is highly effective, excessive use may be at the expense of the quality of the text. Iyyer et al. used back-translation to generate adversarial examples that are syntactically similar. The authors trained an encoder-decoder model that takes a sentence and a target syntactic structure as input, and produces an adversarial sentence. This adversarial sentence is a paraphrase of the original sentence. Sentences can be modified structurally or by adding, deleting or substituting specific words. Alzantot et al. used a black-box population-based genetic optimization algorithm to generate semantically and syntactically correct examples. Their aim was to attenuate the amount of modified words between the original and adversarial examples to take care of grammatical and semantic correctness.

Original Text: Tyrannosaurus rex achieved its massive size due to an enormous growth spurt during its adolescent years.
Adversarial Text: Tyrannosaurus rex achieved its massive size due to an enormous growth spurt durnig its adolescent years.

Figure 2.1: Modified text from the FastWordBug adversarial framework by Goodman et al.

2.2 Sentiment Analysis

Due to the growth of internet usage, online opinions are accessible and used extensively in measuring the general approval of certain products, ideas and organizations. The amount of reviews found online is enormous, and extracting useful information from them requires analysis. This is only possible in practice by automation, according to Liu. Sentiment analysis is the extraction of information with regards to the tonality and sentiment of corpora. For example. determining whether a review is positive or negative is part of sentiment analysis. Although the term sentiment analysis is often perceived to

be the categorization of text as either positive or negative, which is called polarity detection, there is more to it. Other sentiment analysis tasks include word sense disambiguation, anaphora resolution, sarcasm detection, metaphor understanding and aspect extraction. [4].

In this study, the focus will be on polarity detection. As stated, this task consists of classifying for a given text whether the expressed opinion in a document or a sentence is positive, negative, or neutral [3]. The current study will use two classifiers trained to detect polarity in the movie reviews from the IMDB dataset. The polarity of a document is also called its opinion orientation and denotes the general feeling expressed by opinions. Opinionated documents or texts contain certain opinion words, which are words that are commonly used to express positive or negative sentiment. Algorithms can be used to find such opinion words and construct suitable features for a classifier. Sentiment analysis classification is after all build on these features.

2.2.1 Classification for Sentiment Analysis

Methods used for sentiment analysis in machine learning can be categorized in two main classes: supervised and unsupervised classification. the former of which is more common and is used in the current research. Pang et al. showed a first successful attempt of training an effective supervised polarity classifier for movie reviews on the IMDB dataset, outperforming humans.

In supervised classification the model uses a training set and a test set. The training set is used for training a model by the examples sentences which are all labeled with their sentiment. In the case of sentiment analysis, the model is learned how to classify each text based on its features and its label, in the training phase. To see how the model is performing, it can be evaluated with the test set. The test set is a set of examples which are new for the classifier (i.e. have not been used in the training phase) and for which the classifier has to predict the label. When using supervised classification for sentiment analysis, a text is represented by a so-called feature vector. A feature vector, in general, is an n-dimensional vector of numerical features that represent some objects [10].

In sentiment analysis there are two main approaches for extracting features. A common approach is by a so-called bag-of-words representation of the text, which makes a vocabulary of all the words in the document with their occurrence frequency. Relative structure is abandoned, and each word is denoted by its frequency in the vector. In this procedure, one can choose to build a two-word pair vocabulary, which is called a bi-gram vocabulary. In general, a group of n consequent words is called an n-gram.

A more recent method for feature extracting is word embedding. While in the bag-of-words approach each word is represented by a large vector that is necessarily sparse (i.e. containing a lot of zeros), word embedding maps words to a high-dimensional vector space that can be represented in a compact way. Each word is then represented as a numerical vector, and words that are similar are mapped close together in the embedded space [27]. Word embedding can be learned as part of fitting a neural network on text data, which provides a representation of words and their relative meanings to each other.

2.2.2 Finding Features for Sentiment Analysis

According to Koncz and Paralic, "Proper feature selection techniques in sentiment analysis have got a significant role for identifying relevant attributes and increasing classification accuracy.". Pang et al. showed that the feature denoting whether a uni-gram is present is useful to incorporate. In their study, selecting just the uni-grams resulted in the highest accuracy for the Naive Bayes classifier as well as support vector machine models. Other features such as relative position, bi-grams and adjectives were less effective. Although selecting uni-grams as features had the best result, selecting just adjectives and part-of-speech also yielded a decent accuracy. Chenlo and Losada found that the combination of certain features is promising as well. Features such as uni-grams or bi-grams combined with sentiment lexicon as well as part-of-speech tags consistently give good performance for subjectivity classification.

Part-of-speech tagging is a natural language processing pre-processing technique. It is of great value for the understanding of the meaning of sentences since it can be used to extract relationships between words and build knowledge graphs. In Figure 2.2 all words in the sentence are provided with their POS tag. Additionally, a list of each possible POS tag for each word is added.

The_DT first_JJ time_NN he_PRP was_VBD shot_VBN in_IN the_DT hand_NN as_IN he_PRP chased_VBD the_DT robbers_NNS outside_RB ._.

first	time	shot	in	hand	as	chased	outside
JJ	NN	NN	IN	NN	IN	JJ	IN
RB	VB	VBD	RB	VB	RB	VBD	JJ
		VBN	RP			VBN	NN
							RB

Figure 2.2: Sentence with POS-tags, source: "Màrquez et al. 2000, table 1."

Part-of-speech tagging can be used to analyze the opinions from a syntactical point of view and while leaving the original syntax of a sentence undisturbed [2]. For sentiment analysis, part-of-speech tagging is useful for extracting features because certain POS tagged words contribute a lot to subjectivity classification. These words carry emotion and are therefore important in deciding whether a certain text is supposed to be of positive or negative nature. Pronouns, adverbs, modals and adjectives are found to contribute the most to sentiment classification [4]. In particular, adjectives were found to carry information about subjectivity [37] [14]. Therefore, these POS tags in particular will be investigated.

Chapter 3

Methods

This chapter lays out the structure of the experiment. The dataset that is used to train the classifiers from Section 3.2.1 and Section 3.2.2 is analysed and explained in Section 3.1. In section Section 3.2.3 the proposed method for generating adversarial examples is made clear. Furthermore, the generated adversarial examples are tested on a small sub-set of the data from Section 3.1.

3.1 Data

For this research two classifiers are trained on the IMDB dataset[1][26]. This dataset consists of 50.000 movie reviews, each classified as either positive or negative (binary). The dataset contains an even number of positive and negative reviews. Only highly polarized reviews are included, the neutral reviews with a score higher than 4 and lower than 7 are left out. The average length of a review is 200 words. The test set contains 120.064 words and the train set contains 11.437.783 words. This dataset was chosen, due to its highly polarized nature. This characteristic benefits the process of checking if the generated adversarial examples actually flipped over the labels completely. Figure 3.1 provides two examples of review fragments for both labels (positive and negative). Note that the reviews in the example are not yet preprocessed. Moreover, both classifiers will train on 49.500 reviews and are tested on 500 reviews.

[1] http://ai.stanford.edu/~amaas/data/sentiment/

"I thought this was a wonderful way to spend time on a too hot summer weekend,
sitting in the air conditioned theater and watching a light-hearted comedy.
The plot is simplistic, but the dialogue is witty and the characters are likable
(even the well bread suspected serial killer). ... "

"Some films just simply should not be remade. This is one of them.
In and of itself it is not a bad film. But it fails to capture the flavor and the terror
of the 1963 film of the same title. Liam Neeson was excellent as he always is,
and most of the cast holds up, with the exception of Owen Wilson, who just did not bring
the right feel to the character of Luke. ..."

Figure 3.1: Fragments of two reviews in the IMDB dataset: positive (top) and negative (bottom)

3.1.1 Preprocessing

To clean the dataset, the data will undergo a few changes. First the stopwords are removed from the dataset, for which the stop word set of nltk is used. Nltk is a toolkit of python libraries for natural language processing.[2] Pak and Paroubek found that a frequent use of modal words indicate a positive opinion in a text. This is why, for experiment purposes, the set of nltk is altered by removing the words with the POS-tag *modal*, since these words are of value for sentiment analysis. Examples of modals include: could, may or must. The part-of-speech tags are subtracted from the Spacy pipeline *en_core_web_sm* [3] . Furthermore, the reviews are cleaned of HTML tags and URLs, also including all punctuation. Although exclamation marks are important indicators of sentiment, it was chosen to still leave them out since the performance of the classifiers was decent without. All the text is converted to lower case, otherwise the classifiers would, for example, see "The" and "the" as two different tokens.

3.2 Classifiers

3.2.1 Naive Bayes

Naive Bayes has a proven history of yielding high accuracy in text classification as well as being a good feature extractor [36]. The Naive Bayes classifier works by calculating the probability of different characteristics of the data, associated with a specific class. This follows the theorem named after Thomas Bayes:

$$\mathrm{P(A \mid B)} = \frac{P(B|A),P(A)}{P(B)}$$

Often, the conditional probability in the form of P(effect | cause) [4] is known, e.g. the probability of a patient having a stiff neck when diagnosed with meningitis. However, the other way around, the probability of a patient having meningitis when they have a stiff neck, depends on many factors. As stated by Russell and Norvig, "The use of direct causal or model-based knowledge of the theorem of Bayes provides the crucial robustness needed to make probabilistic systems feasible in the real world."

[2] https://www.nltk.org/
[3] https://spacy.io/usage/linguistic-features
[4] P(B|A)

The classifier tries to figure out the underlying distribution of the data: how the data is generated. For this, Naive Bayes is categorized as a generative model. To simplify the learning process of the classifier, Naive Bayes assumes that the semantic features of a sentence are independent of each other [32]. In this study, the multinomial Naive Bayes classifier from scikit-learn was used.[5]

The reviews are vectorized by the TF-IDF vectorizer. TF-IDF stands for *term frequency-inverse document frequency*. TF-IDF, essentially, is a weight metric which determines the importance of the words for a particular document. The value is calculated by the term frequency of a word in a certain document, and adjusted by the inverse document frequency, since some words appear more frequently in general. The inverse document frequency is calculated by taking the total number of documents and dividing it by the number of documents that contain a word, and calculating the logarithm. After applying the metric, words that appear often in most documents score lower, since they are less meaningful [30]. As for the IMDB dataset, applying the TF-IDF vectorizer results in a low score for words such as film, movie, story and character.

3.2.2 Text Based Convolutional Neural Network

A convolutional neural network (CNN) is a deep artificial neural network that uses some special layers. A CNN has normal hidden layers and other layers called convolutionial layers that have filters which are able to detect patterns. Most CNNs also use pooling layers, and are widely used in image classification [19]. The patterns of an image can be, for example, edges, shapes or objects. An edge-filter, for example, in a convolutional layer, would be able to detect edges in an image. The deeper we go in the neural network, the more sophisticated what these filters detect becomes. The use of CNNs is not restricted to the field of image classification. It can be applied to text classification with the help of word embedding. Instead of pixels, the vectors in the rows of the matrix now corresponding to a word. Kim and Jeong, showed that a simple CNN with little hyper-parameter tuning and static vectors can perform excellent on sentence classification. Lai et al. reported that a CNN may even capture the semantic of texts better compared to recurrent neural networks (RNN). As stated by the author, "A CNN can select more discriminative features through the max-pooling layer and capture contextual information through convolutional layer."[22] Just as RNN's, Convolutional Neural Networks are able to capture sequential patterns, doing so by applying the convolutional filters. Moreover, a CNN can be faster than a RNN in computation time. In the study of Yin et al., a CNN turned out to be 5 times faster than a RNN. It is for these reasons that this study will use and test CNNs.

As stated before, CNNs use layers that are applied to local features. Before applying these, the data is usually padded[6] and processed by an embedding layer. To pad the data sequences, zeros are added for a maximum length of 120. The embedding layer is initialized with random weights and learns an embedding for all of the words in the training dataset. After the embedding layer, a convolutional layer is used as a filter to slide over the input vectors, which represent words, and extract the important features. The stride[7] value is set to 1 . To reduce the output dimensionality, a global maximum pooling layer is added. The output of the global maximum pooling layer is a single array of length 128. The output contains the maximum values which summarize the presence of the feature, thus keeping the most salient information. Additionally, two fully connected layers are added with a

[5] https://scikit-learn.org/stable/
[6] It is required that all inputs have the same length
[7] A parameter that modifies the amount of movement over the rows in the tensor

Rectified Linear Unit (ReLU) and Sigmoid activation function consecutively. The architecture of the layers is as follows: 1) Embedding layer, 2) Convolutional layer, 3) Global Maximum Pooling layer, 4) Dense layer 1, 5) Dense layer 2. The flow of the model is shown in Figure 3.2 as well as the shapes of the input and output of the layers. To implement the convolutional neural network the Sequential model from the Keras library was used [8].

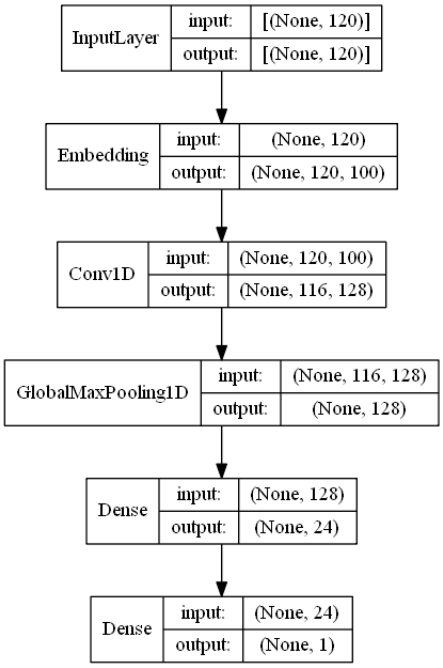

Figure 3.2: Architecture and flow of the CNN layers

[8] https://keras.io/guides/sequential_model/

3.2.3 Part-of-speech Focused Adversarial Examples

Adversarial examples were generated to analyse the robustness of the two classifiers. These examples are black-box, as the characteristics of both classifiers will not be taken into consideration while creating the examples. The examples were generated by substituting POS-tags with their synonyms. These tags are adjectives, adverbs and modals. Their corresponding nltk POS-tags are: JJ, JJR and JJS for the adjectives, RB, RBR and RBS for the adverbs and MD for the words tagged with modal. The R and S suffixes stand for comparative and superlative consecutively. An example of the tags JJR and JJS is: prettier and prettiest. An example of the RB and JJ tags can be found in Figure 3.3. Additionally, alternative part-of-speech tags such as just noun and verbs are tested as well as constituent combinations. It is expected however, that selecting too many tags will make the text unreadable and thus that examples with large substitutions would not qualify as a successful adversarial example.

The proposed method generates adversarial examples which will maintain semantic similarity. The sentences should still be readable as only a few words are replaced[9] and the structure of the original review remains the same. Semantic similarity is maintained by substituting the words by their synonyms. These synonyms are chosen randomly from the synonym set of the word. Note that choosing synonyms with preference to frequency would most likely result in stronger adversarial examples, as this way the text would probably have a larger semantic similarity. As in the study by Ren et al., the synonyms that will be the replacements are provided by WordNet [10], which is a large lexical database of the English language. Not all words appearing in the IMDB dataset have a synonym in the set. These words will be ignored and not be replaced. Moreover, some words have their copy present in their synonym list. It is for this reason that in the experiment, in some runs the copies will be removed from the synonym list. This will produce adversarial examples in which the replacement words can not be identical to the original word. The pseudo-code of the algorithm described before is given in Algorithm 1. An example

Algorithm 1: POS based synonym substitution

```
Data: Synonymset, Text, Postag(s)
Result: Adversarial example
foreach word in Text do
    if word.postag == Postag then
        synonyms ← Synonymset(word);
        if synonyms not empty then
            word ← random.choice(synonyms)
        end
    end
end
```

of a fragment of a modified review can be seen in the 3.3. This is a fragment of a successful adversarial example, since both classifiers originally gave the output *negative* but classified this review as *positive* after modification. To add to that, a human assessor would still believe the review in Figure 3.3 to be a negative review. In the fragment, the adjectives and adverbs are replaced. Note that although

[9] As just the words from a small selection of POS tags are replaced
[10] https://wordnet.princeton.edu/

the replacements are semantically similar they are not semantically indifferent from the original words.

To check if the adversarial examples are sufficiently similar, the present author checks the first 20 generated modified reviews of each experiment run. If the semantics of a replacement word has changed too much, this is reported as in Appendix B. Since only 20 modified reviews for each experiment are checked, in this study, a limit is set of at most 5 out of 20 modified reviews that can be faulty in an experiment run. Moreover, a modified review in which more than the chosen limit of 20% of the replaced words lost too much of their semantic, is considered faulty. Note that as the replacement words are chosen at random from the synonym list, each run has slightly different results. Thus this check is not strictly conclusive.

"... The only benefit to this movie was that it's so astonishingly bad, you do get a few laughs out of it. The really scary thing is that a majority of the people watching the movie with us seemed to enjoy it. Creepy. ..." **Classified as negative**

" ... The only benefit to this movie was that it's so **amazingly regretful**, you do get a few laughs out of it. The **rattling chilling** thing is that a majority of the people watching the movie with us seemed enjoy it. Creepy. ..." **Classified as postive**

Figure 3.3: Example of a successful adversarial attack, original (top) and adversarial example (bottom)

3.3 Experiments

For clarity, the experiments in this study are listed below.

- POS-tag(s): Adjectives, Classifier: NB, Identical substitution: False
- POS-tag(s): Adjectives, Classifier: NB, Identical substitution: True
- POS-tag(s): Adjectives, Classifier: CNN, Identical substitution: False
- POS-tag(s): Adjectives, Classifier: CNN, Identical substitution: True
- POS-tag(s): Adjectives, Modals, Classifier: NB, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Classifier: NB, Identical substitution: True
- POS-tag(s): Adjectives, Modals, Classifier: CNN, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Classifier: CNN, Identical substitution: True
- POS-tag(s): Adjectives, Modals, Adverbs, Classifier: NB, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Adverbs, Classifier: NB, Identical substitution: True
- POS-tag(s): Adjectives, Modals, Adverbs, Classifier: CNN, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Adverbs, Classifier: CNN, Identical substitution: True

- POS-tag(s): Adjectives, Modals, Adverbs, Nouns, Classifier: NB, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Adverbs, Nouns, Classifier: NB, Identical substitution: True
- POS-tag(s): Adjectives, Modals, Adverbs, Nouns, Classifier: CNN, Identical substitution: False
- POS-tag(s): Adjectives, Modals, Adverbs, Nouns, Classifier: CNN, Identical substitution: True

3.4 McNemar Test

The McNemar statistical significance test was used in order to check whether the performance of the classifiers on the adversarial examples differs. Typically, the test is used to compare two different classifiers. However, the test may also be used to compare the same classifier on two different test sets. Since the model can only be evaluated once on the test set with adversarial examples, this test is the only test with acceptable Type-1 error [7]. A type-1 error means that the classifier validates a statistically significant difference even though there isn't one.

The McNemar test gets its results from a contingency table as can be seen in Table 4.2 and Table 4.3. The test's statistic relies on the fact that both classifiers are trained on exactly the same training data. This applies for the current study, since the same classifier is compared for different test sets, i.e. the original and adversarial reviews. In the contingency table, the classifier tested on the original reviews and the classifier tested on the modified reviews will be compared by the amount of correct and incorrect outputs. The McNemar statistic will be based on the bottom left and top right cell values. Those values represent the amount of outputs one classifier has correct, which the other has incorrect and visa versa. These values check whether the disagreements between the two test-cases match or not.

The statistic is reporting on the different correct or incorrect predictions between the two classifiers. The statistic of the McNemar test is calculated as follows: Statistic $= \frac{(Yes/No - No/Yes)^2}{(Yes/No + No/Yes)}$.
In which Yes/No is the value in which classifier 1 has the correct prediction but classifier 2 has the incorrect answer (No/Yes works the other way around) [8]. The null hypothesis entails that the two models disagree to the same amount. If the p value $<= 0.05$ the null hypothesis will be rejected. If the null hypothesis is rejected, this suggests that there is evidence that the two cases disagree in a different way. In this study, this would imply that the classifier predicts the reviews in a different way when they are modified.

Chapter 4

Results

Both the Naive Bayes classifier as well as the CNN are trained on the training set consisting of 49500 unique reviews and tested on the original test set as well as the modified test set of 500 reviews. This modified test set has been altered by the synonym-substitution algorithm. The classifiers are tested on accuracy to determine the success of the adversarial examples. Accuracy is formally defined by the following formula:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

The accuracy metric works poorly with a class-imbalanced dataset. However, the IMDB dataset has an equal number of positive and negative reviews. A large decrease in accuracy after testing on the modified reviews implies that the classifier makes more mistakes on the adversarial examples and thus that labels have been flipped to the incorrect output. This would show that the adversarial examples are successful. Table 4.4 shows the performance of the classifiers. The classifiers are tested on adversarial examples with altering POS tags selected to be replaced. Moreover, a distinction is made between the algorithm always selecting a replacement unequal to the original word and those examples in which this is not the case. The former will be referred to and abbreviated as *neq* and the latter as *eq*. As can be seen in the results from table 4.4, both classifiers show a decrease in accuracy for all the tested adversarial examples. Note that even though both classifiers have around the same value in accuracy on the original test set, 0.846 for NB and 0.842 for the CNN respectively, the score for the CNN drastically decreased on the altered test set. A decrease of 16.8pp can be seen when the POS tags md + adv + noun + adj have been selected, while the highest value of decrease in accuracy for Naive Bayes is 4pp when selecting md + adv + adj neq.

From the results of the similarity check in Appendix B, it can be concluded that only the adversarial examples with the POS tag combinations Adj eq, Adj neq, Adj + Md eq, Adj + Md neq and Adj + Md + Adv eq are sufficient semantically similar. The generated reviews from the other combinations have changed too much. The adversarial examples are tested on significance. In tables 4.2 and 4.3 the contingency tables are reported of both the NB and the CNN when replacing adjectives, adverbs and modals. Since this combination performs the best while the semantic similarity is preserved. The p values are reported together with the decrease in accuracy in table 4.4. From the contingency tables the McNemar test results in stat = 8, p = 0.041 for the Naive Bayes classifier

and stat = 26, p = 0 for the CNN. Both p = 0.041, p = 0 < 0.05, so the null hypothesis will be rejected.

POS-tag(s)	Accuracy NB	Decrease NB	P value	Accuracy CNN	Decrease CNN	P value
Adj eq	0,828	1,8	0,152	0,788	5,4	0,052
Adj neq	0,836	1	0,405	0,778	6,4	0,098
Md + adv + noun + adj eq	0,818	**2,8**	0,136	0,734	10,8	0
Md + adv + noun + adj neq	0,808	**3,8**	0,026	0,674	**16,8**	0
Md + adv + adj eq	0,826	2	0,041	0,736	10,6	0
Md + adv + adj neq	0,806	**4**	0,001	0,748	9,4	0
Md + adj eq	0,836	1	0,424	0,77	7,2	0,098
Md + adj neq	0,838	0,8	0,557	0,788	5,4	0,009

Table 4.1: Overview of the performance of the classifiers for each combination of POS tags

	Original correct	Original incorrect
Modified correct	395	8
Modified incorrect	28	69

Table 4.2: The contingency table for the Naive Bayes classifier

	Original correct	Original incorrect
Modified correct	357	26
Modified incorrect	62	49

Table 4.3: The contingency table for the convolutional neural network

The accuracy would probably decrease further when substituting more words, since the difference between original and modified reviews would be larger. Table 4.4 shows that adding the POS tag *noun* to the already existing set of adjectives, adverbs and modals indeed gives a larger decrease in accuracy 2,8pp eq, 3,8pp neq for Naive Bayes and 10,8pp eq, 16,8pp neq for the CNN respectively. The generated adversarial examples however, from which 20 have been checked by the present author, were not sufficiently semantically similar to the original reviews, as can be seen in Appendix B, probably caused by a high level of ambiguity of nouns. This high ambiguity could result in a synonym set with a high semantic variance. 20 out of the 20 checked modified reviews with the POS tag *noun* were not sufficient semantically similar. In figure 4.1, an example of such a modified example can be seen. Note that some nouns have been replaced with a word with a very different (although still related) meaning, for example *romp* has been replaced by *tomboy*.

Moreover, as stated before, adjectives contribute the most to a sentence opinion orientation and also appear considerably often in reviews. Would it not be sufficient to modify only the adjectives? This was tested on both classifiers, substituting all words tagged with either JJ, JJR or JJS. This resulted in an accuracy difference of -1,8pp eq, 1,0pp neq for NB and -5,4pp eq -6,4pp neq for the CNN

" ... All in all it's full of excuses to dismiss the film as one overblown pile of junk. Stallone even managed to get out-acted by a horse! However, if you an forget all the nonsense, it's actually a very lovable and undeniably entertaining romp that delivers plenty of thrills, and unintentionally, plenty of laughs. ..." **Classified as negative**

" ... All in all it's full of excuses to dismiss the **cinema** as one **grandiloquent peck** of **debris**. Stallone even managed to get out-acted by a **cavalry**! **Nonetheless**, if you forget all the **gimcrack**, it's **in truth** a very **jazzable** and undeniably entertaining **tomboy** that delivers **slew** of thrills and **accidentally**, **slew** of laughs ..." **Classified as positive**

Figure 4.1: Example of a review in which the nouns, adjectives, adverbs and modals have been replaced

respectively. These adversarial examples did not show significant differences with the McNemar test as can be seen in Table 4.4.

It would be interesting to find out which POS tag contributes the most to the decrease in accuracy after modifying the reviews. To analyse this, each prominent POS tag, including adjectives, adverbs, nouns, verbs and modals, is individually selected and tested. Because some POS tags appear more in a text, this is accounted for by dividing the decrease in accuracy by the probability that a certain word has that specific POS tag. This will be called the *contribution ratio* and can, for each classifier, be seen in Table 4.4. It is interesting that the two classifiers both have different POS tags contributing the most to the decrease in accuracy. Moreover, in the absence of other overall trends, the adjectives and modals score high for both classifiers. Interesting is, that modals occur the least in a text but have a high contribution value. Note that the contribution ratio values of one classifier should not be compared to those of the other since the values calculated relatively to the other values of the same classifier.

	Acc	Diff	POS/word	Ratio
Adjectives NB	0,828	**-1,8**	0,134	**13,43**
Adverbs NB	0,844	-0,2	0,08	2,5
Nouns NB	0,834	-1,2	0,271	4,42
Verbs NB	0,842	-0,4	0,0757	5,28
Modals NB	0,844	-0,2	0,0126	**15,87**
	Acc	Diff	POS/word	Ratio
Adjectives CNN	0.788	**-5.4**	0.134	40.2
Adverbs CNN	0.820	-1.2	0.08	27.5
Nouns CNN	0.776	**-6.6**	0.271	24.35
Verbs CNN	0.810	-3.2	0.0757	**55.4**
Modals CNN	0.836	-0.6	0.0126	**47.6**

Table 4.4: Overview of the contribution of the POS-tags to the decrease in accuracy

Chapter 5

Discussion

The contents of this research were focused on the generation of adversarial examples for sentiment analysis. To answer the research question *How can adversarial examples for the NLP task sentiment analysis be generated?*, a method of substituting certain part-of-speech tags with their synonyms is proposed in order to modify the original reviews. As stated prior, adjectives, adverbs and modals seem to contribute the most to a text opinion orientation. For this reason, the choice was made to test whether substituting the words with these specific POS tags results in successful adversarial examples. This study considered both a Naive Bayes classifier and a text based convolutional neural network. The results show that Naive Bayes mis-classified 4pp more reviews that were modified by the algorithm (replacing adjectives, adverbs and modals), which is a substantial amount. Although Naive Bayes is a popular machine learning model for sentiment analysis, these results suggest that the classifier is not robust with regards to the POS based adversarial examples.

As for the text based convolutional neural network, its accuracy dropped by 10,6pp. This is a big and unexpected decrease. One cause for this large decrease in accuracy could be the use of word embedding. The word embedding of the CNN is learned on the fly from the words in the training set. It is likely that some of the replacement words from the synonym list did not appear in the training set. This could cause the CNN to not recognize the synonyms and therefore mis-classify the adversarial examples. This would imply that the proposed synonym substituting method for creating adversarial example is highly effective for neural networks using word embedding. To defend these classifiers to these attacks, CNN's could be trained on even larger sets. The training set could be augmented with reviews in which synonym-substitution has been applied. Additionally, if the CNN were to use an already fixed word embedding this problem might not occur, since this pre-trained word embedding is trained on much larger datasets. For example the Word2Vec pre-trained embedding is trained on 100 billion words, while the dataset used in this study (IMDB) has only 11.437.783 words in total.

Some shortcomings of this study will now be discussed. Because of limited computation resources, the choice was made to use a test set of only 500 reviews. Although a clear difference in accuracy can be seen from our results, a larger dataset would show an even more accurate calculation of the models performance. However, most probably, the size of the test set in this study did not affect the results too much. Notable is that the McNemar test used for testing significance, is a rather conservative test, meaning that the test will always maintain the chance of rejecting the null hypothesis well underneath

the significance importance level. Possibly, some tested adversarial examples are falsely designated as not significant. However, the results show that most p values from examples other than the chosen adversarial examples were above p=0.130, which shows a very weak evidence against the null hypothesis. Another thing to point out is that the contribution to the decrease in accuracy by the modals for the Naive Bayes classifier might be questionable. The fact that a lot of modals appeared in the original stopword list suggest that these words occur a lot in all documents. Although they were removed from the stopword list, their high frequency could possibly make them less important for the classification by the TF-IDF vectorizer. This would suggest that the modals would not contribute much to the decrease in accuracy. In the literature however, it is found that modals appear a lot in positive texts [28]. Moreover, modals could contribute a lot to the sentiment of movie reviews specifically since from the results of this study, this POS tag had one of the largest contribution to the decrease in accuracy of the Naive Bayes classifier.

Current time state-of-the-art models for natural language processing such as BERT are of the transformer archetype. These so-called transformers are multiple layers of neural-network constructions that assign vectors to words and subwords in different contexts which the model finds in the training data [6]. The models that are discussed and experimented upon in this study are context-independent. Meaning that they do not use user or domain metadata (context) to adapt the predictions [16]. Since the proposed method for generating adversarial examples in this paper operates on word-level, not making use of the context, it could be that context-aware models such as BERT are more robust to these examples. This would imply that the proposed method perhaps could not generalize well to the current state-of-the-art classifiers. It is however, that the proposed method changes words in every sentence (if the words correspond with the selected POS tags) throughout the whole review, which could make changes in the context. Thus the advantage of the context-aware models could be limited.

There were a few further restrictions in this research that mandate discussion. It was previously discussed that adversarial examples for natural language tasks could take multiple forms. For example, the perturbations can be on character level or consist of modified word location. In this study, the choice was made to maintain semantic similarity in modifying the reviews. An adversarial example is successful if humans can still correctly classify the reviews without being fooled by the modifications. As a consequence of limited resources, a human survey was not achievable but would be far more accurate and reliable than the sample checked by the present author. Moreover, it seems that the algorithm does not always chooses a replacement word that has exactly the same meaning as the original word. The synonyms supplied by WordNet do not take word-ambiguity into account. This can make some sentences look odd, and occurs mostly when substituting nouns. The proposed method of generating adversarial examples with replacing adjectives, adverbs and modals produces modified reviews that are sufficiently semantically similar. Therefore, in this study, although the replacement words will be semantically similar to the original words, they will not always be semantically indifferent.

5.1 Conclusion

This study proposes a black-box un-targeted method for generating adversarial examples for the natural language processing task sentiment analysis. The results from this research suggest that the Naive Bayes and the convolutional neural network model archetypes are vulnerable to the proposed approach on generating adversarial examples. The CNN might be more robust to the proposed method if it uses a pre-trained word embedding trained on a very large dataset. For the proposed method, for both

NB and the CNN, the best combination in terms of adversarial example generation of selected POS tags is adjectives combined with adverbs and modals. The results show that this selection lowers the performance of the classifiers the most while preserving the semantics of the reviews the best. Although the method produces successful adversarial examples, it can not be concluded that the modified reviews are completely semantically indifferent. A stricter synonym-set could improve the proposed method. The results of this study laid bare weaknesses in the tested machine learning models by generating successful adversarial examples. It could prove to be a stepping stone for further research into the topic of adversarial examples for natural language processing. Additionally, the proposed method may be used to generate more data examples to augment data sets and thereby improve the accuracy of machine learning models.

5.2 Future Work

This research showed that certain machine learning algorithms are vulnerable to adversarial examples. By training the classifiers with these generated adversarial examples, future work could aim to defend the classifiers against certain adversarial attacks and improve the overall model accuracy. It would be important to stress semantic similarity of the generated adversarial example and it would be beneficial to create strict synonym sets with synonyms that are semantically indifferent. This could help make the modified text example indistinguishable semantically to human observers. Moreover, the use of the TF-IDF vectorizer caused uncertainty on the contribution of the modals to the decrease in accuracy. To provide clarity, in future studies other vectorization approaches would have to be used. The proposed black-box method for generating adversarial examples can be used to spot weaknesses in other machine learning models as well. Furthermore, in future research the proposed method can be tested on generalisability, as this was not addressed in the current study.

Bibliography

[1] Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1316. URL https://www.aclweb.org/anthology/D18-1316.

[2] Dr. Muhammad Asghar, Aurangzeb Khan, Shakeel Ahmad, and Fazal Kundi. A review of feature extraction in sentiment analysis. *Journal of Basic and Applied Research International*, 4:181–186, 01 2014.

[3] Erik Cambria, Dipankar Das, Sivaji Bandyopadhyay, and Antonio Feraco. *A Practical Guide to Sentiment Analysis*. Springer Publishing Company, Incorporated, 1st edition, 2017. ISBN 3319553925.

[4] Erik Cambria, Soujanya Poria, Alexander Gelbukh, and Mike Thelwall. Sentiment analysis is a big suitcase. *IEEE Intelligent Systems*, 32(6):74–80, 2017. doi: 10.1109/MIS.2017.4531228.

[5] José M. Chenlo and D. Losada. An empirical study of sentence features for subjectivity and polarity classification. *Inf. Sci.*, 280:275–288, 2014.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

[7] Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput.*, 10(7):1895–1923, October 1998. ISSN 0899-7667. doi: 10.1162/089976698300017197. URL https://doi.org/10.1162/089976698300017197.

[8] Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10:1895–1923, 1998.

[9] Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. Black-box generation of adversarial text sequences to evade deep learning classifiers. *CoRR*, abs/1801.04354, 2018. URL http://arxiv.org/abs/1801.04354.

[10] Paul Ginsparg, Paul Houle, Thorsten Joachims, and Jae Hoon Sul. Mapping subsets of scholarly information. *Proceedings of the National Academy of Sciences of the United States of America*, 101 Suppl 1:5236–40, 05 2004. doi: 10.1073/pnas.0308253100.

[11] Zhitao Gong, Wenlu Wang, Bo Li, Dawn Song, and Wei-Shinn Ku. Adversarial texts with gradient methods, 2018. URL http://arxiv.org/abs/1801.07175.

[12] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. 2014. URL http://arxiv.org/abs/1412.6572. cite arxiv:1412.6572.

[13] Dou Goodman, Zhonghou Lv, and Minghua Wang. Fastwordbug: A fast method to generate adversarial text against NLP applications. *CoRR*, abs/2002.00760, 2020. URL https://arxiv.org/abs/2002.00760.

[14] Vasileios Hatzivassiloglou and Janyce M. Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. In *COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics*, 2000. URL https://www.aclweb.org/anthology/C00-1044.

[15] Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. Adversarial example generation with syntactically controlled paraphrase networks. *CoRR*, abs/1804.06059, 2018. URL http://arxiv.org/abs/1804.06059.

[16] Aaron Jaech and Mari Ostendorf. Low-rank RNN adaptation for context-aware language modeling. *Transactions of the Association for Computational Linguistics*, 6:497–510, 2018. doi: 10.1162/tacl_a_00035. URL https://www.aclweb.org/anthology/Q18-1035.

[17] Hannah Kim and Young-Seob Jeong. Sentiment classification using convolutional neural networks. *Applied Sciences (Switzerland)*, 9, 06 2019. doi: 10.3390/app9112347.

[18] Peter Koncz and Jan Paralic. An approach to feature selection for sentiment analysis. *INES 2011 - 15th International Conference on Intelligent Engineering Systems, Proceedings*, 06 2011. doi: 10.1109/INES.2011.5954773.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems*, 25, 01 2012. doi: 10.1145/3065386.

[20] Volodymyr Kuleshov, Shantanu Thakoor, Tingfung Lau, and Stefano Ermon. Adversarial examples for natural language classification problems, 2018. URL https://openreview.net/forum?id=r1QZ3zbAZ.

[21] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016. URL http://dblp.uni-trier.de/db/journals/corr/corr1607.html#KurakinGB16.

[22] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. Recurrent convolutional neural networks for text classification, 2015. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9745.

[23] Bing Liu. Sentiment analysis and opinion mining. volume 5, 05 2012. ISBN 978-3-642-19459-7. doi: 10.2200/S00416ED1V01Y201204HLT016.

[24] Xiaodong Liu, Hao Cheng, Pengcheng He, Weizhu Chen, Yu Wang, Hoifung Poon, and Jianfeng Gao. Adversarial training for large neural language models. *CoRR*, abs/2004.08994, 2020. URL https://arxiv.org/abs/2004.08994.

[25] Yanpei Liu, Xinyun Chen, Chang Liu, and Dawn Song. Delving into transferable adversarial examples and black-box attacks. *CoRR*, abs/1611.02770, 2016. URL http://arxiv.org/abs/1611.02770.

[26] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. pages 142–150, June 2011. URL https://www.aclweb.org/anthology/P11-1015.

[27] Amit Mandelbaum and Adi Shalev. Word embeddings and their use in sentence classification tasks. *CoRR*, abs/1610.08229, 2016. URL http://arxiv.org/abs/1610.08229.

[28] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. volume 10, 01 2010.

[29] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. pages 79–86, July 2002. doi: 10.3115/1118693.1118704. URL https://www.aclweb.org/anthology/W02-1011.

[30] Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181, 07 2018. doi: 10.5120/ijca2018917395.

[31] Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1085–1097, 2019.

[32] I. Rish. An empirical study of the naive bayes classifier. Technical report, 2001.

[33] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, December 2002. ISBN 0137903952. URL http://www.amazon.ca/exec/obidos/redirect?tag=citeulike09-20&path=ASIN/0137903952.

[34] Lichao Sun, Kazuma Hashimoto, Wenpeng Yin, Akari Asai, Jia Li, Philip S. Yu, and Caiming Xiong. Adv-bert: BERT is not robust on misspellings! generating nature adversarial samples on BERT. *CoRR*, abs/2003.04985, 2020. URL https://arxiv.org/abs/2003.04985.

[35] Shixin Tian, Guolei Yang, and Ying Cai. Detecting adversarial examples through image transformation. In *AAAI*, 2018.

[36] Christos Troussas, Maria Virvou, Kurt Junshean Espinosa, Kevin Llaguno, and Jaime Caro. Sentiment analysis of facebook statuses using naive bayes classifier for language learning. In *IISA 2013*, pages 1–6. IEEE, 2013.

[37] Peter D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. *CoRR*, cs.LG/0212032, 2002. URL http://arxiv.org/abs/cs/0212032.

[38] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of CNN and RNN for natural language processing. *CoRR*, abs/1702.01923, 2017. URL http://arxiv.org/abs/1702.01923.

[39] Xiaoyong Yuan, Pan He, Qile Zhu, Rajendra Rana Bhat, and Xiaolin Li. Adversarial examples: Attacks and defenses for deep learning. *CoRR*, abs/1712.07107, 2017. URL http://arxiv.org/abs/1712.07107.

Appendices

Appendix A

Stopwords

The words from the stopword set used in the preprocessing of the data, are listed below. The stopword set is provided by nltk but has been adjusted by removing words that are tagged as *modal*, as discussed in Section 3.1.1.

Against, ve, now, and, haven, don, here, have, needn, having, being, other, I, had, don't, than, haven't, so, has, with, o, off, weren, down, no, because, y, over, at, as, hadn, when, of, doing, aren't, in, was, who, same, very, again, by, t, on, or, were, to, isn't, doesn, wasnt, why, the, that, is, but, above, weren't, where, more, such, didn, after, how, before, d, into, not, below, for, most, didn't, under, does, won, while, hasn't, are, about, until, ain, through, been, from, can, then, between, there, ma, shan, hasn, aren, out, once, up, did, m, too, during, if, an, only, do, that'll, am, nor, re, hadn't, be, further, just, a, youve, doesn't, what, isn, wasn.

Appendix B

Check for Semantic Similarity

To check if the adversarial examples are sufficiently similar, the first 20 generated modified reviews of each experiment are checked. In this study, a limit is set of at most 5 out of 20 modified reviews from an experiment run that can be faulty. In this study, a modified review will be considered faulty if more than 20% of the replaced words lost too much of their semantic. The results ar shown in Table B.1. Note that as the replacement words are chosen at random from the synonym list, each run has slightly different results.

	Semantically similar	Semantically different	Approved
Adj eq	17	3	True
Adj neq	17	3	True
Adj + Md eq	15	5	True
Adj + Md neq	16	4	True
Adj + Md + Adv eq	13	7	False
Adj + Md + Adv neq	15	5	True
Adj + Md + Adv + Noun eq	0	20	False
Adj + Md + Adv + Noun neq	0	20	False

Table B.1: Semantic similarity check for each adversarial example combination