Haters Gonna Hate Me

Anzidei, M.
Department of Data Science and Knowledge Engineering
Maastricht University

Bongrand, P.
Department of Data Science and Knowledge Engineering
Maastricht University

1 Introduction

Since people are using more and more social media, research on security on online social network has grown substantially in the last years. This leads to an increase in number of hateful activities that exploit such platforms. Furthermore, social media are criticized for not doing enough to prevent hate speech on their sites and have come under pressure to take action against hate speech. Our goal is clear: prevent any type of harassment on social media. Therefore, we decided to create an automated hate speech and offensive comment detection. It would analyze any type of comment or tweet written in english and classify it into 3 categories: hate speech, offensive and neutral. The idea behind this is to ban any user than does spread Hate Speech messages, and send a warning message to anyone who is offensive over a person, group or community.

2 Literature

The choice of hate speech detection derives from two extremely interesting research papers: "Hate Speech Detection Using Natural Language Processing Techniques" [1] and "Automated Hate Speech Detection and the Problem of Offensive Language" [4] The first paper goal is to look at how Natural Language Processing applies in detecting hate speech. Furthermore, this paper also applies a current technique in this field on a dataset. As neural network approaches outperforms existing methods for text classification problems, a deep learning model has been introduced, namely the Convolutional Neural Network. The second paper is focusing on the same task, except that it is using a multi-class classifier to distinguish between these different categories. The authors find that Tweets without explicit hate keywords are also more difficult to classify. An interesting challenge would be to be able to identify those hate speech tweets that do not contain hate keyword and evaluate and compare the different approach explored in those papers.

3 Dataset

The dataset [5] consists of 25'000 labelled tweets: 19'190 offensive comments, 4'163 normal comments and 1'430 hate speech comments. The dataset is perfect to train the algorithm, recognizing hate speech and offensive comments. The first column of the dataset refer to the comments and the second one categorizes each comment in hate speech (0), offensive (1) and not offensive/neutral (2). As a rule of thumb we divided this dataset into 2 subsets: Train and Test, the train subset contains 80% of the comments.

4 Features

We cleaned the data by removing: punctuation, links, numbers, contraction of words, whitespaces and specific characters derived from twitter such as: mentions, Re-tweet, Hash-tags. We chose not to lowercase each tweet, since uppercase letters can also imply a meaning in terms of hate speech and offensiveness.

5 Language Models

In order to estimate the relative likelihood of different phrases we used two different Language Models.

5.1 Word2Vec

Word2vec [8] is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words in that corpus. Given enough data, usage and contexts, Word2vec can make highly accurate guesses about a words meaning based on past appearances.

Those guesses can be used to establish a words association with other words. Since, we have a training dataset of 20'000 tweets, we consider it way enough in order to apply the Word2Vec Language Model to our dataset.

5.2 Bag of Words (BOW)

A bag-of-words model [6], is a way of extracting features from text for use in modeling. A bag-of-words is a representation of text that describes the occurrence of words within a document. It involves two things: A vocabulary of known words; and A measure of the presence of known words.

5.3 Differences

The main difference is that Word2vec produces one vector per word, whereas BoW produces one number (a wordcount). Word2vec is great for digging into documents and identifying content and subsets of content. Its vectors represent each words context, the ngrams of which it is a part. BoW is a good, simple method for classifying documents as a whole. Therefore, both approaches will highlights different aspects of our dataset.

6 Classifiers

Once we have transformed the tweets into for example vectors, we need to actually classify the tweets among the 3 categories previously described: Neutral, Offensive, Hate Speech. We are doing this using different type of classifiers.

6.1 Decision Trees: Extreme Gradient Boosting (XGB)

One of the many way to classify those tweets is through decision trees. However, a simple decision tree would either be too much based on our training sample and do overfitting. Or, it would be very inaccurate and would just be a bit better than random guesses. Those reasons motivated us to use a decision tree boosting. The XGBoost library [2] implements the gradient boosting decision tree algorithm. XGBoost is very commonly used for supervised learning problems, where we use the training data (with multiple features) to predict a target variable. The idea of this library is that it will make ensemble of tree. It will do so, by adding weak trees (that measure different features) together in order to make better decision at classifying tweets among our three categories.

6.2 Multinomial Naive Bayes

Naive Bayes [7] is a family of algorithms based on applying Bayes theorem with a strong naive assumption, that every feature is independent of the others, in order to predict the category of a given sample. They are probabilistic classifiers, therefore will calculate the probability of each category using Bayes theorem, and the category with the highest probability will be output. Naive Bayes classifiers have been successfully applied to many domains, particularly Natural Language Processing, which motivated us to apply it to tweets classification. The simple design of Naive Bayes classifiers make them very attractive for such classifiers. Moreover, they have been demonstrated to be fast, reliable and accurate in text classification.

6.3 Support Vector Machine (SVM)

An Support Vector Machine [3] is a large-margin classifier: it is a vector space based machine learning method where the goal is to find a decision boundary between two classes that is maximally far from any point in the training data (possibly discounting some points as outliers or noise). It is usually used for Natural Language Processing tasks such as text classification.

7 Results and Analysis

Given the high density of offensive tweet in the dataset, and as all figures highlight it is observable that our algorithm is extremely accurate at identifying offensive expression. Due to the lack of hate speech and neither comments that can be observed on Figure 1. both models clearly fails at distinguishing hate speech tweets from offensive ones.

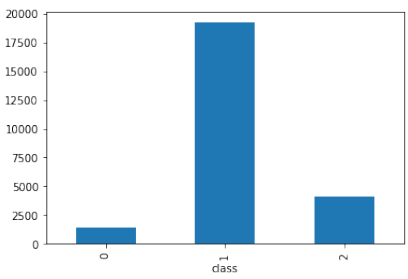

Figure 1: Dataset proportion

Due to this feature of the dataset, the provided accuracy is not highly reliable, and to compute the real quality of the classifier, a deeper analysis of

the result needs to be done. This analysis, will allow us to have a better understanding of the error of our classifiers. The question being: are our classifiers too sensitive to offensive expression or not enough?

To visualize the results of the predictions of each classifier, the algorithm creates automatically a confusion matrix with the percentages of true/false positive and true/false negative. The accuracy of each classifier is provided by a default method which calculates the "score".

7.1 Bag of words

Results provided by each classifier of the algorithm Bag Of Words.

7.1.1 Multinomial Naive Bayes

Starting from the Multinomial Naive Bayes classifier, it can be seen that the accuracy of identifying offensive tweet is really high. As mention previously, the classifiers fails at identifying hate speech with an percentage of well identified hate speech of 3.6%. Moreover, the classifier still identifies more than 80% of hate speech tweets as offensive, which can be seen as a positive feature. Regarding the neutral tweets, the classifier identifies 61% of the actual neutral comment as neutral.

Figure 2: Multinomial NB using Bag Of Words

7.1.2 SVM

Regarding the Support Vector Machine classifier, it is observable that most of the offensive tweets have been classified correctly. Still, neither this classifier can distinguish with a high accuracy between hate speech and offensive comments. Furthermore, this classifier identifies almost perfectly neutral tweets ($\approx 90\%$).

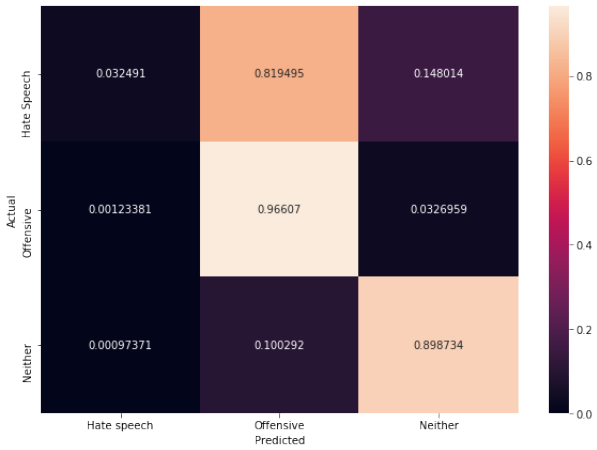

Figure 3: Support Vector Machine using Bag Of Words

7.1.3 Score and Comparison

The table and the previous analysis of both matrices show that the Support Vector Machine is better at classifying the different tweets while using BOW. The true positive values of the MultinomialNB classifier is 3% higher than the one provided by the SVM. But, the true positive values of neutral tweet provided by the SVM is much higher than the one provided by the MultinomialNB by a difference of ($\approx 30\%$). Even though, the accuracy rate is not highly reliable, the difference between the scores represents correctly the analysis of the confusion matrices (Figures 2, 3). Therefore, we can conclude that using Bag of words, the Support Vector Machine stands out as being the best classifier.

Classifiers	**BOW**
MultinomialNB	0.88
SVM	0.91

Table 1: Accuracies of classifiers using Word2Vec

7.2 Word2vec

Results provided by each classifier of the algorithm Word2Vec.

7.2.1 Extreme Gradient Booster

The XGB classifier using Word2Vec identifies 95% of the actual offensive tweets as offensive. Regarding the hate speech tweets, the results look very similar as the other classifiers with the only difference that the correct classified tweets are only 1%.

Figure 4: Support Vector Machine using Word2Vec

7.2.2 Support Vector Machine

The Support Vector Machine classifier using Word2Vec identifies almost every offensive tweets ($\approx 97\%$). However, this classifier clearly does not succeed at identifying hate speech comment, since it identified 0 tweets as containing hate speech. In comparison to the classifiers used by BOW, the true positive Diagonal of the Confusion Matrix of this classifier are the lowest values.

Figure 5: Support Vector Machine using Word2vec

7.2.3 Score and Comparison

In this case is observed that the scores reported in Figure 2 are the same. However, the confusion matrices are slightly different. Therefore, it means that the values of the confusion matrix balance themselves and reach an identical accuracy. Moreover, the XGB classifier is a better classifier because, it first recognize better hate speech ($\approx 1\%$ against $\approx 0\%$ for SVM), and second does wrongly identifies hate speech as neutral text but in a less extent way ($\approx 19\%$ against $\approx 21\%$ for SVM).

Classifiers	**Word2Vec**
XGB	0.85
SVM	0.85

Table 2: Accuracies of classifiers using Word2Vec

7.3 General Comparison

It can be recognized that the result of the confusion matrices are similar for both Language Models. Still each classifier has its own features and is easily recognizable that BOW performs better then Word2vec, for this specific data-set. Both classifier that use BOW have a higher accuracy then the ones used by Word2vec. Finally it can be assumed that the SVM classifier is the one with the best performances.

LM/Classifier	**Word2Vec**	**Bag of Words**
XGB	0.85	...
Multinomial	...	0.88
SVM	0.85	0.91

Table 3: Accuracies of language models using different classifiers

8 Conclusion

Thanks to the analysis it can be assumed that the algorithm is extremely accurate in identifying bad languages. Which implies that every offensive, and hate speech tweets will be identified as bad. Due to the features of the dataset, the combination of BOW and SVM (which is the combination that performs the best), still doesn't distinguish between the offensive tweets and the hate speech ones, categorizing almost any hate speech comment as an offensive one. This feature does not implies directly less quality by the use of the algorithm, to solve real world problems. The program can identify almost perfectly a bad language comment from a neutral one, where hate speech and offensive comments are both seen as bad language content. Further work could be done in the direction, the goal being: Identifying correctly every offensive and hate speech tweets while identifying as few as possible neutral tweets as offensive or hateful.

References

[1] Shanita Biere. Hate speech detection using natural language processing techniques. *Vrije Universiteit Amsterdam*, pages 1–29, 2018.

[2] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. *CoRR*, abs/1603.02754, 2016.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995.

[4] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515, 2017.

[5] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Hate speech and offensive language dataset. https://github.com/t-davidson/hate-speech-and-offensive-language/blob/master/data/labeled_data.csv, August 2017. Accessed on 2019-27-05.

[6] Zellig S. Harris. Distributional structure. *WORD*, 10(2-3):146–162, 1954.

[7] M. E. Maron. Automatic indexing: An experimental inquiry. *J. ACM*, 8(3):404–417, July 1961.

[8] Tomas Mikolov, G.s Corrado, Kai Chen, and Jeffrey Dean. Efficient estimation of word representations in vector space. pages 1–12, 01 2013.