Classification of Hate Speech and Offensive Language

Zhe Sang
zhs46@pitt.com

Zhaomin Xiao
zhx36@pitt.edu

ABSTRACT

Hate speech, can be frequently seen on the social media, has been a malignant cancer to web space that not only influence the relationship between different groups of people but also undermines the stability of society, especially harms mental health for individuals. Yet it is difficult to differentiate hate speech from offensive language, who is less severe and sometimes is a normal phenomenon in people's daily conversation[9]. We address this problem in Twitter tweets with natural language processing and machine learning techniques by finding latent features of users' tweets and classify them into three category.

KEYWORDS

Logistic Regression, Feature Selection, Deep Learning, Cross-validation, Text Classification.

1 INTRODUCTION

Social media like Twitter, Facebook created a broad space on the Internet where users have free access to a large volume of information and meanwhile are nearly unlimited to express their opinion. Freedom of speech, protected by law, grants people of the right to hold opinions without any interference, yet not any speech (e.g., hate speech) can be free speech. Abuse of hate speech imposes enormous harm not only to the mental health of victims but also the stability of society. Generally hate speech is defined as attacking persons or groups of disadvantage, usually by racism, sexism, homophobia, and transphobia, and so on.

Although Facebook and Twitter have conducted a set of policies and rules to prevent users from expressing hate speech, hardly would people be punished by publishing a hate speech, mostly due to lack of support from relevant laws. Moreover, the boundary between hate speech and offensive language, which is less severe and harmful, is ambiguous. Using offensive words in conversation with friends can achieve a number of desirable effects that helps promote social cohesion as well as making humors through self-deprecation and sarcasm. There is no universal standards about what statement will be regarded as offensive, let alone hate speech. Thus, it is still far from getting rid of hate speech thoroughly and in the meantime to a large extent allowing people making humors. It is worthwhile studying techniques that can automatically detect hate speech among a large amount of data.

We study this problem in Twitter since Twitter is a hub that connects people and provides a space for people to communicate, who comes from a broad range of countries, races and social groups. In this situation, Twitter is a place where collisions among different people happen frequently. In our case, the data we used are manually classified into 3 categories: hate speech, offensive language and neither. We explored the data with dimension reduction techniques to get an intuitive insight, assessed several features extracted from tweets. We proposed to compare bagging and boosting algorithms with single machine learning model(e.g logistic regression, SVM) and analyze the potential reason that makes hate speech detection so challenging.

Conference'17, July 2017, Washington, DC, USA
2018. ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORK

Previous researchers has studied this problem quite deeply, from initially simple surface features like n-gram based bag of words model, frequency of URL, mentions, punctuation and tweet length to sentiment analysis, linguistic features (part-of-speech tag) and distributed word representation[18]. Both class machine learning algorithm and deep learning[20] can be used to tackle such problem. For machine learning methods, the most commonly used algorithms are support vector machine [1] and logistic regression [3]. With appropriate parameter tuning and feature engineering, SVM and logistic regression can also achieve a pleasing result. For neural network techniques, Mikolov et al. [14, 15] used Word2Vec to represent word visa word embedding. One year later, Pennington et al.[16] used GloVe to represent word and obtain higher accuracy than before. In terms of building neural network, bidirectional long-short-term-memory recurrent neural network (LSTM) [8] is purposed to take the long term memory and short term memory into consideration, which can solve the problems caused by long-term dependency in vanilla RNN. Besides, gated recurrent units [2] are purposed to be an alternative to the bidirectional LSTM. Additionally, Yoon Kim successfully applied convolutional neural network (CNN) on text classification[10].But since the cost of training model is comparatively huge, traditional machine learning classification algorithms are still commonly used.

3 DATA ANALYSIS

The dataset we used in this project was from *Hate Speech And Offensive Language*[4] (https://github.com/t-davidson/hate-speech-and-offensive-language), which was obtained through Twitter API and filtered based on n-gram (word/phrase) identified by *hatebase.org* and then randomly sampled to build a corpus of 25k tweets. Dataset has been labeled into three class (0: hate speech, 1: offensive language, 2: neither) by *CrowdFlower* (CF) workers, there is a manual for workers which contains some rules that can be used when determining which category a tweet may belong to.

This dataset is highly imbalanced, as shown in Figure 1, where tweets labeled as hate speech only take up less than 10 percent and neither ones take up less than 20 percent.

4 FEATURES

We used several features to build the model to distinguish hate speech from offensive language. The main features consist of POS n-gram, word-based n-gram. Both two features are represented as

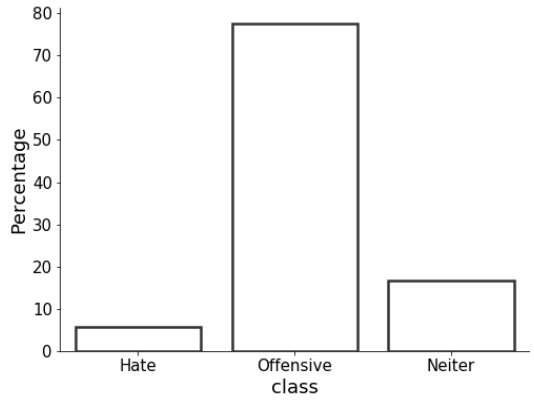

Figure 1: Distribution of 3 classes.

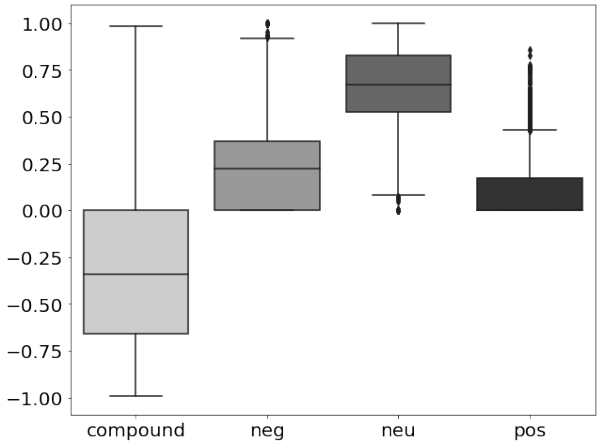

Figure 2: Sentiment score distribution.

TF-IDF matrix. Apart from main features, we also tried to use some additional features in order to obtain higher accuracy.

4.1 Main Features

- **Word n-gram** Word-based unigram, bigram, and trigram are the most important features for this task. Since the most important criterion to determine whether a tweet contains offensive language or hate speech is to find if this tweet contains some specific words.

4.2 Additional Features

- **POS n-gram** POS refers to part of speech. unigram, bigram, and trigram can also used to determine if one tweet contains bad information.
- **Sentiment Analysis** Sentiment analyzer is used to give each tweet four sentiment scores, which can reflect the possible attitude of the author. The sentiment labels include '*neg*', '*pos*', and '*compound*'. In order to measure the degree of negativity, we use the following formula.

$$score = (2 * neg - pos - compound) * 10$$

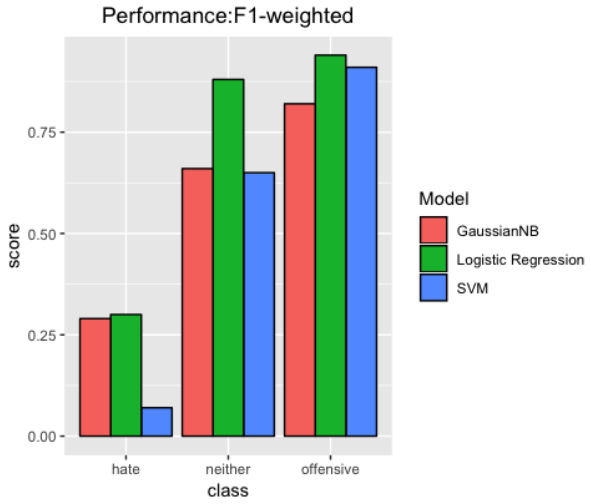

Figure 3: weighted F1 of each class.

- **Retweet Indicator** Whether the tweet is retweeted or not is also considered.
- **Function Words** Function words are words that have little lexical meaning but express grammatical relationships among other words within a sentence. We use a list of 34 function words.
- **Punctuation** Different languages use punctuation differently.
- **Length of tweet** We use the length of cleaned tweet, in terms of the number of characters and the number of words in each Tweet.
- **Capital Letters** The number of capital letters are also related with the prediction.

5 CLASSIFICATION METHODS

5.1 Classic Machine Learning Methods

Spam email detection is the most common text classification task, which is composed of bag-of-word model based word representation and Naive Bayes classification model and proved to be effective. Here we follows similar path to do our classification task.

Original texts were transformed with a set of processing methods including stemming, tokenization, feature extraction and construction of TF-IDF matrix. Operations applied on texts bring high dimensionality and sparsity that makes it difficult to train models and may cause overfitting. We consider using Logistic Regression with ℓ_1 penalty to do automatic feature selection before data were fed to model. After feature selection, we tested logistic regression, SVM, and Naive Bayes with default parameter setting to simply compare performance s of different models. We find that logistic regression generally yields the best result, as shown on Figure 3. We then focused on testing the impacts of different features and tuning parameters of model.

Feature	$F_{1-macro}$
$Word(uni, bi, tri-gram)$	0.68
$POS(uni, bi, tri-gram)$	0.41
$SentimentNegativityScore$	0.40
$Num\_Capitals$	0.33
$Num\_Terms/Chars/Punctuations/functions$	0.29
$Retweet$	0.29

Table 1: Independent performance of feature types detailed in 4.1 and 4.2.

Model	$Precision_{macro}$	$Recall_{macro}$	$F_{1-macro}$
Baseline	0.67	0.69	0.68
LR_{-FS}	0.73	0.77	0.75
LR_{+FS}	**0.74**	**0.81**	**0.77**
LSTM	0.81	0.73	0.76
LSTM + Attention	0.81	0.71	0.77
CNN + LSTM	0.74	0.62	0.66

Table 2: Performances of models.

5.2 Deep Learning Techniques

In terms of deep learning techniques, here we tested 3 neural network structure: bidirectional LSTM[7], LSTM with Attention mechanism[19] and CNN with LSTM. Texts were embedded using pre-trained word2vec model from GloVe Twitter[17].

5.2.1 Bidirectional LSTM. The embedding layer has the dimension of 200 and static word weight. LSTM has the output dimension of 30. Then additional auxiliary features(Sentence length, number of punctuations, number of URLs, number of mentions, number of hashtags) are combined with LSTM output. Next, data will go through a fully connected layer of 256 nodes. The final output layer is composed of 3 nodes in order to generate 3 probabilities of 3 classes.

5.2.2 Bidirectional LSTM with Attention. This network structure is similar to the above. The only difference is to add an additional attention layer after LSTM layer.

5.2.3 CNN with Bidirectional LSTM. The main purpose of CNN here is to help capture features of embedded text. 1 dimensional convolutional layer with number of filter of 64 and kernel size of 5 is applied to embedded data. After convolution, 1 dimensional max pooling layer with pool size of 4 is applied. What follows next is similar as above, they are LSTM layer, fully connected layer and a final output layer.

The deep learning model is built under framework of Keras and we referred sample codes provided by Keras. Dropout is used to prevent network from overfitting. The dropout rate is 0.25 for each layer. The activation functions of two hidden layers are rectifier and sigmoid function. The optimization method is Adam. The metrics of the network is accuracy. The neural network is trained by i5 CPU.

6 EVALUATION

We evaluated the predictive power of several features independently with the logistic regression model. Instead of using weighted measurement, we use unweighted method because the dataset is imbalanced, the score could be high if we predict all test samples to the class with large proportion on training dataset.

As expected, bag-of-word based word representation is the most predictive feature among the rest, which is reasonable as the concurrence of words reflects the meaning of a sentence to some extent. It's fairly robust to use bag-of-word model to represent sentence meaning in the field of classic machine learning.

Part of speech follows next in terms of $F_{1-macro}$ score. POS belongs to the lexical feature, which reflects the grammar structure of a sentence. We assume that the expression of hate speech and offensive language should be different and pos is a possible solution to unveil such difference.

Sentiment score yields similar performance with POS. Sentiment analysis produces 4 numerical values for each text: *compound*, *neg*, *neu*, and *pos*. We do not use *neu* since the amount of neutral information is not important to distinguish hate speech from offensive language. Since we want to equally weigh *neg*, *pos*, and *compound*, and we subtract the *pos* and *compound* from *neg* to emphasize the difference between the amount of negative information and the sum of positive information and compound information, we should multiply *neg* by 2. And we also want to stress the weight of *neg* and make final score more important, then we multiply *2 * neg - pos - compound* by 10. This idea leads to the formula in 4.2.

Rest of features that were tested proves to be little predictive when used independently.

These independent features do not accurately indicate the importance regarding producing right classification. Strong predictive feature such as bag-of-word model absolutely is the key, but proper combination of several weak predictive features may together contribute to the final result.

To find the best combination of features, we use the TF-IDF based word representation as our main feature and intuitively add additional features and compare the performance. As Table 2 shows, the baseline model uses logistic regression without penalty and the input feature is unigram-based TF-IDF without feature selection, the performance is fair. The LR_{-FS} is the model using logistic regression with ℓ_2 penalty and the input features are the features that we finally decide to use. The model structure of LR_{+FS} is same as LR_{-FS}, but the input features are automatically selected from all features we were to use. Features selection contributes a lot to the increase of recall without sacrificing precision and F1-score, which is the relative higher coverage. And this is what we want. Our result, shown in Fig 4, is similar to the reported one.

For deep learning approach, we have provided surface features of text as auxiliary input apart from embedding of text for all 3 models. CNN text classifier behaves worst, one reason could be impropriate network structure another could be that CNN in our case can not capture significant features among 3 classes. Directly processing embedded text with LSTM seems to be helpful as both LSTM and LSTM+Attention yield better result than CNN text classifier. However, such models tend to classify more hate speech as

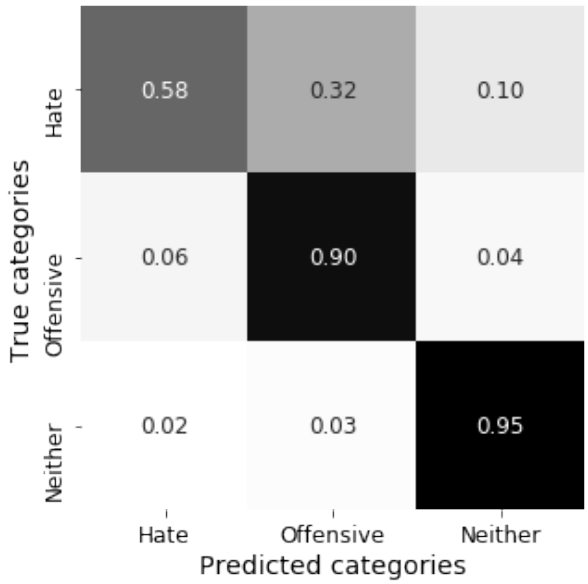

Figure 4: Weighted recall of each class.

offensive language that generate higher precision score but lower recall score, although they generate same F1 score as our LR_{+FS}.

7 DISCUSSION

After all these trials and errors, we have learned a lot from both features and models.

We have actually tested many features, including the number of URLs, Mentions, and Hashtags; character based n-gram; lemmatized-based preprocessed approach; the number of syllables and misspelling in tweet, etc. These features sound promising but does not help in the task of identifying hate speech and offensive language, they may be strong predicators in other text classification tasks.

We have also tried to reduce the dimensionality of feature space with PCA, but the result is not pleasing. A valid way to do dimension reduction could be to allow less word to be put in the word-of-bag model, that is to limit the maximum and minimum document frequency of a term as well as limit the size of bag. The reason why it works is probably we avoid overfitting and introduce more generalization by reserving only the terms having significant effect on result.

Feature selection also plays an important role in our classification task. Truth of matter is that we can always tune the model by altering the complexity and penalty to achieve similar best results with different features input, but the key feature is always bag-of-word word representation. Such method is of robustness to text classification. Those fancy linguistic or lexical features may help, but is not significant as for now. We would expect linguistics researchers to discover more features that could be used in text mining.

Identification of hate speech and offensive language not doubt is a tricky problem, even human can not well distinguish them without understanding the context. In our experiment, there are 108 tweets that are misclassified as hate speech by our best model. However, when we take a look at these 108 tweets, more than half of tweets have been considered to be hate speech by at least one reviewer. People are subjective when judging these tweets, there is no golden rules to objectively assess texts. The ground truth is not totally correct.

8 FUTURE WORK

With the development of deep learning [12], we have more choices for text classification tasks. Besides of RNN, convolutional neural network, introduced by LeCun et al. [13], works very well in image classification tasks. Because of parallel computing technique and the development of hardware, advanced GPU can accelerate the model training very much, which shows its promising future.

There are also much progress in optimization methods, besides of the most common choice, SGD, many variants of SGD are introduced. Adagrad [6], Adam [11], and Nadam [5] are all good choices.

Although deep learning techniques become more and more popular recent years, classic machine learning methods are still very powerful. Actually, for most tasks, the best results of classic machine learning methods are very close to that of deep learning techniques.

In addition to further research on deep learning model. Another perspective is to use under-sampling, over-sampling approach to deal with highly skewed dataset. Alternatively, an augmentation approach to increase the minority class is to translate text to another language and then translate them back. It is quite interesting method that we would probably in the future studying.

REFERENCES

[1] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. 1992. A Training Algorithm for Optimal Margin Classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory (COLT '92)*. ACM, New York, NY, USA, 144–152. https://doi.org/10.1145/130385.130401

[2] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 1724–1734. https://doi.org/10.3115/v1/D14-1179

[3] D. R. Cox. 1958. The Regression Analysis of Binary Sequences. *Journal of the Royal Statistical Society. Series B (Methodological)* 20, 2 (1958), 215–242. http://www.jstor.org/stable/2983890

[4] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *arXiv preprint arXiv:1703.04009* (2017).

[5] Timothy Dozat. 2015. Incorporating Nesterov Momentum into.

[6] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *J. Mach. Learn. Res.* 12 (July 2011), 2121–2159. http://dl.acm.org/citation.cfm?id=1953048.2021068

[7] A. Graves and J. Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM networks. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, Vol. 4. 2047–2052 vol. 4. https://doi.org/10.1109/IJCNN.2005.1556215

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation* 9 (12 1997), 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735

[9] Timothy Jay. 2009. Do offensive words harm people? *Psychology, public policy, and law* 15, 2 (2009), 81.

[10] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[11] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).

[12] Yann LeCun, Y Bengio, and Geoffrey Hinton. 2015. Deep Learning. *Nature* 521 (05 2015), 436–44. https://doi.org/10.1038/nature14539

[13] Yann LeCun, Bernhard E. Boser, John S. Denker, Donnie Henderson, R. E. Howard, Wayne E. Hubbard, and Lawrence D. Jackel. 1990. Handwritten Digit Recognition with a Back-Propagation Network. In *Advances in Neural Information Processing Systems 2*, D. S. Touretzky (Ed.). Morgan-Kaufmann, 396–404. http://papers.nips.cc/paper/293-handwritten-digit-recognition-with-a-back-propagation-network.pdf

[14] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013).

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 3111–3119. http://dl.acm.org/citation.cfm?id=2999792.2999959

[16] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–1543.

[17] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[18] Anna Schmidt and Michael Wiegand. 2017. A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*. 1–10.

[19] Sergey Zagoruyko and Nikos Komodakis. 2016. Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. *CoRR* abs/1612.03928 (2016). arXiv:1612.03928 http://arxiv.org/abs/1612.03928

[20] Steven Zimmerman, Udo Kruschwitz, and Chris Fox. 2018. Improving Hate Speech Detection with Deep Learning Ensembles.. In *LREC*.