"Share Love not Hate" - Assessing Hate Speech Detection methods using TF-IDF and POS tagging approach on Twitter

Kshitija Hande
Lakehead University - Computer Science
khande@lakeheadu.ca

***Abstract*— The spread of social networks and their unfortunate use for hate speech -direct attacks towards a group or an individual based on their actual or perceived aspects of identity, such as ethnicity, religion, and sexual orientation- automatic detection of hateful text has become a pressing problem causing personal attacks, online harassment and bullying behaviours. Hate speech detection on Twitter is significant for applications like controversial event extraction, AI chatbots, content recommendation, sentiment analysis, etc. We define this task as being able to classify a tweet as hateful, offensive or neutral.**

This task is challenging because of the complexities in natural language constructs. in this work, we replicate existing detection systems which uses TF-IDF and POS tagging with unigram, bigram and, trigram feature extraction and Logistic Regression as feature selector and Linear SVM as classification of tweets into hateful, offensive language and neither, achieving best F1-score of 0.91.

***Index Terms*— hate speech, offensive language, classification, Twitter, toxic language, TFIDF, n-gram, bag-of-words, POS tagging, linear SVM, logistic regression**

I. INTRODUCTION

Micro-blogging websites and online social networks are attracting internet users a lot more than any other kind of websites. Services offered by Twitter, Facebook and Instagram are more and more popular among people from a variety of backgrounds, cultures and interests. Their contents are growing rapidly, an interesting example of the big data. Big data have been attracting the attention of researchers, who are interested in the automatic analysis of people's opinions and the structure/distribution of users, etc. While these websites offer an open platform for people to discuss and share their thoughts and opinions, this makes it almost impossible to control the nature of content.

Taking advantage of this, people with different backgrounds, cultures and believes, tend to use aggressive and hateful language. Nowadays, with growth of online social networks, and increasing conflicts happening around the world, the censorship of content remains a controversial topic, dividing people into two groups, one supporting it and the other opposing it. It is even easier to spread such trends among young as well as older generations, with respect to other "cleaner" speeches. For these reasons, Burnap and Williams [1] collecting and analyzing temporal data allows decision-makers to study the escalation of hate crimes following "trigger" events. However, official information regarding such events is scarce given that hate crimes are often unreported to the police. Social networks in this context present a better and more rich, yet untrustworthy and full of noise, source of information. To overcome noise and the non-reliability of data, we require an efficient way to detect both hateful and, offensive posts in data collected from social networks.

We must define toxic language in order to tackle this issue. We broadly segregate toxic language into two categories namely, hate speech and offensive language. As stated by Wikipedia, Hate speech is defined as "any speech that attacks a person or a group based on attributes such as religion, race, ethnicity, gender, gender identity, sexual orientation, or disability". Offensive language can be described as a text that includes abusive slurs or derogatory expressions.

Filtering hateful tweets manually is not scalable, urging researchers to identify automated alternatives. Most of the earlier work revolves either around manual feature extraction or the use of representation learning methods which is then followed by a linear classifier.

In this work, we will focus on the problem of classifying a tweet as hateful, offensive or neutral. The task is challenging due to the intrinsic complexity of the natural language constructs, there are various forms of hatred, hate tweets are targeted towards different targets, the same meaning can be represented in various ways. We address the task of text classification in terms of hateful content, which uses

- a language-agnostic solution, which does not use pre-trained word embedding
- an experiment of the model on a Twitter dataset to find its performance on the classification task.

In the proposed solution we used the Twitter dataset we train our classifier model using n-gram and Term Frequency-Inverse Document Frequency (TF-IDF) for feature extraction. We perform an experimental study on the features obtained using Logistic Regression for selecting features based on importance weights and Linear Support Vector Machine for classification. Our results show that after hyperparameter tuning of n-gram and TFIDF features, Linear SVM is the best performing model.

The remainder of this paper we present recent work along with our experiments and results in detail concluding with future work for this research project.

II. RELATED WORKS

Existing methods mainly project the problem as a supervised document classification task, they are in two categories: one relies on manual feature engineering that are then consumed by algorithms such as SVM, Naive Bayes, and Logistic Regression (classic methods); the other represents the more recent deep learning model that employs neural networks to automatically learn features from raw data (deep learning methods).

Nobata et al. [5] used lexicon, n-gram, linguistic, syntactic, pretrained, "word2vec" and "comment2vec" features to perform the classification task into two classes, and obtained an accuracy equal to 90%.

Some works targeted the detection of hate text in Twitter. Kwok and Wang [6] targeted the detection of hateful tweets against black people using unigram features which gave an accuracy of 76% for binary classification. They identified the focus of the hate speech toward a specific gender, ethnic group, race or other makes the collected unigrams related to that specific group. Therefore, the built dictionary of unigrams cannot be reused to detect hate speech towards other groups with equal efficiency. Watanabe and Ohtsuki [12] used the relation between words along with bag of words (BoW) features to distinguish hate speech.

With the emergence of hate speech and offensive language datasets, numerous studies have touched upon cross-dataset generalization since 2018. Gröndahl et al. [7] trained a range of models, cross-applying them on four datasets. The models included LSTM -one of the most popular neural network in text classification- and CNN-GRU (Zhang et al. [8]), which outperformed previous models on six datasets [9]. Gröndahl et al. [7] experiments show that adversarial training does not completely mitigate the attacks, and using character-level features makes the models systematically more attack-resistant than using word-level features [9].

III. CHALLENGES IN EXISTING RESEARCH

1) It is a challenging task, as analysis of the language in the typical datasets shows that hate speech lacks unique and discriminative features.
2) One possible issue with pre-trained embeddings is Out-Of-Vocabulary (OOV) words, especially on Twitter data due to the nature of tweets. Thus the pre-processing will be done such that it helps to reduce the noise in the language and hence the scale of OOV. For example, by hashtag segmentation, we transform an OOV '#BanPUBG' into 'Ban' and 'PUBG' that are probable to be included in the pre-trained embedding models.
3) A statistical review on an existing annotated dataset of tweets as well as existing research can show the relationship between the user bias in expressing their opinion that is offensive/hateful, and the associated annotation class labels. This points towards the requirement to explore the user features to improve the classification accuracy of a supervised learning system.
4) Although tweets rarely contain two full sentences, splitting long tweets into two has been proved to cause loss of linguistic information hence, we will avoid doing so during pre-processing phase. Instead we propose to train our model by normalizing it over maximum word limit.
5) Training based on domain specific data is expected to increase performance on tasks like hate speech detection. However, the results from previous research showed that there were not any improvements it did not prove to show huge improvements in capturing features. Hence, we will not be training our model on domain-specific corpora.

IV. METHODOLOGY

Based on the results of past work, we decided to extract features using n-gram from the input text and weight them using TF-IDF. We then feed these features to a classification head which will then classify it into three categories: hateful, offensive, and neither.

A. Experimental Setup

We used scikit-learn [18] library in Python for training and experimentation. We used Google Colab for executing our experiments.

B. Dataset

We used a publicly available dataset from the HuggingFace library [20]. This dataset is available on Crowdflower [19], used in [ref 1]. It contains tweets that have been manually classified into the following classes: "Hateful", "Offensive" and "Neither". The tweets were manually coded by CrowdFlower (CF) workers, where they were asked to label each tweet as one of three categories: hate speech, offensive but not hateful, or neither offensive nor hate speech. The tweets in the dataset were compiled with a hate speech lexicon comprising of words and phrases recognized by web users as hate speech, compiled by Hatebase using the Twitter API.

Class labels	count
hateful	840
offensive	11,571
neither	2,458
Total	14,869

TABLE I
CLASS DISTRIBUTION IN TRAINING SET

C. Data Preprocessing

As part of data cleaning and preprocessing, we convert the tweets to lowercase and lemmatize words. We also remove the following undesirable contents from the tweets:

Dataset	count
Training set	14,869
Validation set	3,718
Test set	6,196
Total	24,783

TABLE II
DATASET DISTRIBUTION

- Stopwords
- Irregular Spacing
- URLs
- Twitter Mentions
- Retweet Symbols
- Emojis

We randomly shuffle and split the dataset into train, test and validation, once the dataset is in desired format. We used total 24,783 labelled data examples which is then split into train, validation and test set (Refer Table II).

In train set, the distribution of labels was as described in Table I. The distribution is unbalanced where majority of tweets were labelled offensive and least amount of tweets were labelled as hate speech.

D. Feature Extraction

After data pre-processing we create unigram, bigram and trigram features, each being weighted by its TF-IDF. We also construct Penn Part-of-Speech (POS) tag unigrams, bigrams and trigrams to gain information about its syntactic structure. For hashtags, number of characters, words, and syllables in each tweet, we include binary and count indicators. The formula for TF-IDF where can be defined as follows:

$$tfidf(d,t) = tf(t) \ * \ idf(d,t) \tag{1}$$

where,
tf(t) = the number of times the term(t) appears in the document divided by the total number of words in the document and,
idf(d,t) = log of total documents in a set divided by number of documents containing term(t)

In this work, we limit TF-IDF and POS tag encoded vector size to 10,000 and 5,000 respectively. Increasing size of these vectors further did not show significant improvements in model performance and was computationally expensive.

E. Model

First using Logistic Regression with L2 regularization we reduced the dimensionality of the data. We found that particularly, the Logistic Regression and Linear SVM performed notably well as compared to other baselines like Naive Bayes and Linear Regression.

For final model we used Logistic Regression with L2 regularization performed well in previous papers (Burnap and Williams [1]; Waseem and Hovy [11]) and helped predicting probabilities of each class better than previous experiments. We used it for feature selection by fitting the data, and then transform it. It fits transformer to input X and true label y with optional parameters and returns a transformed version of X which will be used to train Linear SVM.

In SVM, only support vectors has an effective impact on model training, it implies that removing non support vector data-points has no effect on the model. Hence the risk of over-fitting is less as compared to other baselines.

We use a one-versus-rest framework where a separate classifier is trained for each class and the class label with the highest predicted probability across all classifiers is assigned to each tweet.

F. Experiments

1) TF-IDF with unigram, bigram and, trigram feature extraction which is then used to train Linear SVM classifier.
2) Feature extraction using TF-IDF and POS tagging, both with unigram, bigram and, trigram features. We feed these features to Linear SVM classifier for training.

In both experiments, Logistic Regression feature selector and Linear SVM model uses L2 regularization and 0.01 as regularization parameter. Hinge loss creates a boundary that separates negative and positive instances as +1 and -1, with +1 on the right and -1 on the left side of the boundary (Refer to equation 2). In this work, we took squared hinge as loss function.

$$Cost(\hat{y}, y) = \begin{cases} max(0, 1 - \theta^T x) & if, y = 1 \\ max(0, 1 + \theta^T x) & if, y = 0 \end{cases} \tag{2}$$

Linear SVM cost function with L2 regularization can be formulated as follows:

$$J(\theta) = C[\sum_{i=1}^{m} Cost(\hat{y}, y)^2] + \frac{1}{2}\sum_{j=1}^{n} \theta_j^2 \tag{3}$$

where, m = number of samples in dataset and, n = number of features

Logistic Regression which uses λ as the parameter in front of regularized term to control the weight of regularization, similarly, SVM uses C in front of the fit term. In our experiments we tuned hyper-parameter C to optimum value i.e. 0.01 which prevents our model form over-fitting.

G. Evaluation Metrics

We used F1-score to evaluate our model since it is a classification task and our class distribution is imbalanced as described in Dataset section IV-B. We chose F1-score over Accuracy as our main metric because accuracy can be largely attributed by a large number of True Negatives whereas in most business cases False Negative and False

Positives usually pertains business costs (tangible and intangible). Hence F1-score was a better measure to seek balance between Precision and Recall and, since there is uneven class distribution because of a higher number of offensive language examples.

V. Results

The best performing has an overall F1-score of 0.91 which is from experiment 2 taking TF-IDF and POS tagging for feature extraction (Refer Table IV). Using TF-IDF alone resulted in overall F1-score of 0.90 (Refer Table III), hence using POS tagging helped improve the feature extraction process. Furthermore, we can see that precision and recall of hateful class in the best performing model is 0.60 and 0.79 respectively, which is the least across all class predictions. Most of the miss-classification occurred in the hateful class and the least number of tweets are classed as hateful or offensive when their true label is neither.

We confirmed that tweets containing racist or homophobic phrases tend to be tagged as hateful by our model. Some tweets are in response to other hate speakers as well. We also observed that some offensive tweets have been miss-classified as hate because of the vocabulary used although, in a different context. It is noted that not all hate comments make use of curse words but contain a negative sentiment. Our model is also not able to detect types of hate speeches that occur infrequently during training phase as compared to the prevalent ones.

As compared to previous work, our model is able to correctly identify offensive language with highest F1-score of 0.94 (Refer Table IV). With highest precision and recall scores, this class is rarely miss-classified as hateful. Racist and homophobic comments are classed as hateful whereas, sexist or curse words as tagged as offensive. Consistent to finding by Waseem and Hovey [11], human coders appear to consider sexist and derogatory terms against women to be only offensive.

VI. Discussions

Although the tweets in our dataset contain racist and sexist terms, it is plausible that the majority of Twitter users often makes use of them in day-to-day conversations. For example, they tend to use n*gga instead of n*gger as in line with the findings of Kwok and Wang [6]. It is also observed the most users have an inclination to quote lyrics from rap songs which can be wrongly considered as offensive, but when taken into context turn out to be an inside joke. Such type of tweets can also lead our model to wrongly flag them. While our model still miss-classifies some of the offensive tweets as hateful, a vast majority avoids such errors by differentiating the two.

The category with neither hate nor offensive language had the second-best F1 score, the examples in the dataset are non-toxic because the terms are included in the Hate-base lexicon. The miss-classification in this category tend to mention social issues. This also leads us to believe that the user's intention and history of toxic comments can lead our model to make better predictions.

VII. Limitations of proposed solution

1) Small datasets can make our models susceptible to overfitting, and biases in datasets transfer to models, this can be a possible limitation. Additionally, the dataset was highly skewed with fewer hate speech samples as compared to other two classes.
2) A probable limitation can be the effect of tweet normalisation on the accuracy of proposed model.
3) Offensive comments can be made without using explicit toxic language, they can go undetected from our current model.
4) We are not taking context into account while looking for hateful vocabulary. Hence use of such words in a humorous context can be incorrectly flagged.
5) Model is susceptible to over-fitting, this can be addressed by using 10 fold cross-validation.

VIII. Conclusion and Future work

Existing hate speech detection models generalize poorly on new, unseen datasets. This is because of the limits of existing NLP methods, dataset building, and the nature of online hate speech, and are often interrelated. The behaviour of social media users and particularly haters pose added challenge to existing NLP approaches. Small datasets available from Twitter, can make models susceptible to overfitting, and biases in datasets transfer to models. Some biases come from different sampling methods, others show existing biases in our society.

Hate speech evolves with time and context and thus has a lot of variation in expression. Existing attempts to address these challenges span across adapting state-of-the-art in other NLP tasks, refining data collection and annotation, and drawing inspirations from domain knowledge of hate speech. More work can be done in these directions in order to improve generalizability hence in our proposed model instead of using pre-trained models, we will use word-frequency vectorization like TF-IDF and POS tagging which will include users tendency to post hateful text. Our experiments achieved 0.91 F1-score using Logistic Regression feature selector and Linear SVM classifier, which was used to classify tweets as hateful, offensive or neither.

In future, we aim to build a richer dictionary of hate speech patterns that can be used, along with a unigram, bigram and trigram features, including user's tendency to post hateful and offensive online texts. We plan to experiment with state-of-the-art deep learning architectures like LSTM and GRU to increase accuracy. We also plan to use larger datasets from across the web to train and test the classifiers and use them to study the increase and decrease of cyber-hate on online social media platforms.

Class labels	Precision	Recall	F1-score
hateful	0.57	0.76	0.65
offensive	0.96	0.91	0.93
neither	0.78	0.90	0.84
overall			0.90

TABLE III

RESULTS FROM EXPERIMENT 1 USING ONLY TF-IDF FOR FEATURE EXTRACTION

Class labels	Precision	Recall	F1-score
hateful	0.60	0.79	0.68
offensive	0.97	0.92	0.94
neither	0.81	0.93	0.87
overall			**0.91**

TABLE IV

RESULTS FROM EXPERIMENT 2 USING TF-IDF AND POS TAGGING FOR FEATURE EXTRACTION

ACKNOWLEDGMENT

I would like to thank all the researchers who have made their resources available to the research community.

REFERENCES

[1] P. Burnap and M. L. Williams, "Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making," Policy Internet, vol. 7, no. 2, pp. 223–242, Jun. 2015.

[2] W. Warner and J. Hirschberg, "Detecting hate speech on the world wideWeb," in Proc. 2nd Workshop Lang. Social Media, Jun. 2012.

[3] N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati, "Hate speech detection with comment embeddings," in Proc. WWW Companion, May 2015

[4] N. D. Gitari, Z. Zuping, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," Int. J. Multimedia Ubiquitous Eng., vol. 10, no. 4, pp. 212–230, Apr. 2015.

[5] C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang, "Abusive language detection in online user content," in Proc. WWW, Apr. 2016, pp. 140–160.

[6] I. Kwok and Y. Wang, "Locate the hate: Detecting tweets against blacks," in Proc. AAAI, Jul. 2013

[7] Gröndahl, T., Pajola, L., Juuti, M., Conti, M., and Asokan, N., "All you need is" love" evading hate speech detection," in Proceedings of the 11th ACM workshop on artificial intelligence and security (pp.2–12), 2018.

[8] Zhang, Z., Robinson, D., and Tepper, J. , "Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network," in A. Gangemi et al. (Eds.), The Semantic Web (pp. 745–760). Cham:Springer International Publishing. doi: 10.1007/978-3-319-93417-4_48

[9] Yin, W., and Zubiaga, A., "Towards generalisable hate speech detection: a review on obstacles and solutions," in ArXiv, abs/2102.08886, 2021.

[10] Gaydhani, Aditya Doma, Vikrant Kendre, Shrikant B B, Laxmi, "Detecting Hate Speech and Offensive Language on Twitter using Machine Learning: An N-gram and TFIDF based Approach", 2018.

[11] Waseem, Z., and Hovy, D. (2016, June). Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In Proceedings of the NAACL Student Research Workshop. San Diego, California: Association for Computational Linguistics. doi: 10.18653/v1/N16-2013

[12] H. Watanabe, M. Bouazizi and T. Ohtsuki, "Hate Speech on Twitter: A Pragmatic Approach to Collect Hateful and Offensive Expressions and Perform Hate Speech Detection," in IEEE Access, vol. 6, pp. 13825-13835, 2018, doi: 10.1109/ACCESS.2018.2806394.

[13] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma, " Deep Learning for Hate Speech Detection in Tweets," in ¡i¿Proceedings of the 26th International Conference on World Wide Web Companion¡/i¿ (¡i¿WWW '17 Companion¡/i¿), 2017.

[14] Georgakopoulos, S., S. Tasoulis, A. Vrahatis and V. Plagianakos. "Convolutional Neural Networks for Toxic Comment Classification." Proceedings of the 10th Hellenic Conference on Artificial Intelligence, 2018.

[15] Zhang, Z. and Le Luo. "Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter." Semantic Web 10, 2019.

[16] Isaksen, Vebjørn and Björn Gambäck. "Using Transfer-based Language Models to Detect Hateful and Offensive Language Online." WOAH, 2020.

[17] Pitsilis, Georgios K., H. Ramampiaro and H. Langseth. "Detecting Offensive Language in Tweets Using Deep Learning." ArXiv abs/1801.04433, 2018.

[18] Pedregosa, F., et al. "Scikit-learn: Machine learning in Python." Journal of Machine Learning Research, pp 2825–2830, 2011.

[19] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber, "Automated Hate Speech Detection and the Problem of Offensive Language." Proceedings of the 11th International Conference on Web and Social Media (ICWSM), 2017.

[20] Davidson, Thomas and Warmsley, Dana and Macy, Michael and Weber, Ingmar, "Automated Hate Speech Detection and the Problem of Offensive Language", Proceedings of the 11th International AAAI Conference on Web and Social Media, ICWSM '17, 2017.