Sentiment Analysis For Automated Hate Speech Detection

Brent Read
Princeton University
bread@princeton.edu

Abstract

This project implements machine learning methods to determine whether written tweets are considered ”Hate Speech.” By using natural language processing and sentiment analysis methods, we develop a system that can classify text as either hateful, non-hateful, or non-hateful but using offensive language. This work expands upon the existing Hatebase database, which manually labels hateful tweets and sentiment analysis algorithms that can determine whether text is positive or negative in nature. This project finds that algorithms that use linear methods are the most effective, with Naive Bayes as the best performing algorithm.

1 Introduction

Hate Speech is defined as speech that directly attacks a person or group based on features like gender, race, disability or sexual orientation. Hate Speech generally defines a specific attack directed at a well-defined group of people. There is significant debate over the importance of free speech in questions of hate speech, but these matters are beyond the scope of this project. Many European countries have laws that explicitly ban hateful speech or speech that may otherwise incite animosity. In the United States, there are provisions against direct threats, but most forms of conventional hate speech are tolerated under the law.
However, the form of Hate Speech varies wildly between regions and communication medium. Leading up to the Rwandan Genocide of 1994, hateful rhetoric directly contributed to animosity toward the Tutsi people [9]. The radio broadcast contained codified words that described an ”infestation” of ”cockroaches” and other phrases that have subtext that cannot be easily inferred from a literal interpretation of the messages.
Moreover, the colloquial usage of otherwise hateful terms complicates the task of identifying hate speech. Many of these issues are common with sarcasm detection, which has long been a difficult problem natural language processing. However, by using manually labeled data, some of this nuance can be incorporated into a prediction model. Additionally, there are many components of hate speech that are consistent patterns, and certain aspects of tone and structure can perhaps be generalized across different cultures and languages.
Specifically for Twitter, the official website for the company says that ”Twitter prohibits the promotion of hate content, sensitive topics, and violence globally” within advertisements found on Twitter feeds. This system is regulated by users flagging violating ads for manual review. Legally, Twitter makes a distinction between ”hateful conduct” which is explicitly banned, and ”hate speech,” a policy that applies most often to explicit death threats [6]. This project is not predicated on a formal definition for hate speech, but rather seeks to identify Hate Speech in a way that is consistent with the labeled dataset acquired from open data platform CrowdFlower.
The remainder of this project discusses a sentiment analysis platform trained on the open dataset provided by CrowdFlower. Section 2 describes related work in the field of sentiment analysis, and work that has been done to classify hate speech. Section 3 describes the implementation of the system, and the pipeline for data processing. Next, section 4 provides an in-depth look at the machine learning framework used in the final version of the sentiment analysis program. Section 5 provides

an overall assessment of the results of this project, while Section 6 provides future work that can be done to improve this project. The paper concludes with Section 7, which offers concluding remarks on the project as a whole.

2 Related Work

This section will describe work related to sentiment analysis. Section 2.1 will discuss the current state of research in sentiment analysis, and Section 2.2 will discuss research specifically directed at Tweet analysis.

2.1 Sentiment Analysis

Sentiment analysis, otherwise known as opinion mining, uses natural language processing and text analysis techniques to infer subjective information from written text. It has practical uses in appraisal theory and applications toward customer service organizations. This field has grown significantly, as the availability of sentiment data has increased greatly in the recent past. For example, much sentiment analysis research uses user reviews on website like Amazon.com and movie reviews on IMDB.com [7]. At a high level, sentiment analysis attempts to discern text with a positive disposition from text with a negative disposition. Previously classifiers would generally ignore texts that would conventionally be classified as ”neutral” as they are considered to lie close the the boundary condition in a binary classifier, and are thus subject to noisy effects that are beyond the predictive capabilities of the classifier [4]. However, some specific classifiers, like Max Entropy and SVM models can be improved from the introduction of neutral material [4].
Additionally, some work has been done on the processing of Hate Speech found on the Internet. Gerstenfeld et. al looked at material found on 157 extremist website in order to look for patterns [1]. They found that over half of these websites contained racist symbols, such as the logo for the Ku Klux Klan, while a third disavow racism on their websites. Notably, very few of these publications actively encouraged violence. They note that the internet is an effective tool for the solicitation of hatred, and that there are systematic patterns in the ways that these messages are disseminated. Thus, we can ascertain that there may be patterns in hateful tweets that are detectable by some form of classifier.
Detecting hate speech presents a more difficult problem than simple sentiment analysis. Many approaches have used very domain-specific approaches. For example, Warner and Hirschberg used an manually annotated hate speech corpus to train an SVM classifer to identify anti-semetic terms with 94% accuracy [10]. There is significant variance in the ways that these systems are annotated. This system most often fails on hateful terms that are used in a non-hateful way, such as the statement ”Kike is a word often used when trying to offend a jew” [10]. In 2015, Gitari et. al looked at creating a general hate speech detection classifier using a lexicon-based approach. They seek to build a lexicon for hateful phrases, and use an approach that uses contextual features that improve on a simplistic dictionary approach [2]. These methods address some of the issues of the contextualization of hateful tweets, but requires a large training dataset to work. They train a classifier using the Naive Bayes algorithm to create a lexicon. These methods methods are very effective (if not perhaps beyond the scope of this project) but are not fit for working on short data, like tweets. The next section will discuss twitter specific developments for NLP.

2.2 NLP on Tweets

Tweets offer an especially difficult medium for sentiment analysis as they provide noisy data that is necessarily truncated in many cases. Additionally, very few NLP libraries offer support for processing emoji. Saif et. al used SVM to alleviate data sparsity. These methods were effective in dealing with data that has been taken directly off the Twitter firehose, but is not necessary for this project [8]. Many existing linguistic features are not effective on Tweets because they often have their own domain-specific lexicon and rarely have the contextual information that is needed in order to make larger-scale inferences. Kouloumpis et. al found that n-gram models were effective at dealing with Twitter data, while many other conventional NLP methods were not as effective [3]. There are some methods built-in to open source library NLTK that compensate for this type of noisy data which can effectively be used for a project of this type. There have been successful implementations of sentiment detection based on twitter data. This project expand on these methods by using tonal inference and lexical analysis to determine if a given tweet is hate speech.

3 Implementation

This section describes how the hate speech classifier was implemented. Section 3.1 discusses how the dataset was acquired and Section 3.2 discusses how the data was processed.

3.1 Data Acquisition

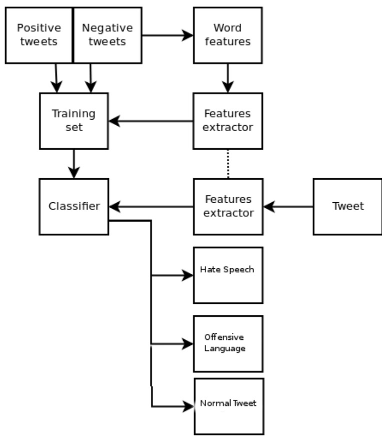
The first part of the data comes from the open data platform CrowdFlower. The dataset includes an anonymized user identification number and the text of their tweet. Each tweet was then rated in one of three categories: ”Contains Hate Speech,” ”Contains Offensive Language, but is not Hate Speech,” and ”Not offensive.” There are 14,509 such responses, which were labeled by the mode response of participants who were paid for their responses. Within this, 2399 were classified as hateful, 4836 were classified as offensive, and 7274 were neither hateful nor offensive. For the purposes of the assignment, there is assumed to be no systematic bias in how responses are reviewed, and are thus considered to reflect the general assessment of hate speech at the time that the polling took place.
The data had already been filtered such that all responses had been labelled in one of the three categories. There was no culling of borderline cases, and each label was given a confidence interval based on the rate that participants rated it in the labeled fashion.
Additionally, experimental data for control were acquired from the NLTK Tweet corpus, which includes a large datasets of tweets that have been taken from the twitter firehose. The Twitter firehose is prohibitively expensive for this particular project, so no current tweets were acquired for inference.
Another source of information for this project came from the online database Hatebase. Hatebase is an online database that catalogs user-reported incidents of hate speech, and provides a definition for them. As the later stages of this project do not use lexical information for the sake of simplicity, this information was largely used for exploratory analysis, as it can determine how people use hate speech, and what groups are targeted in different media.

3.2 Data Processing

The data were read in through plaintext. However, a given tweet can span a variety of uses. Tweets can be retweets of other users, which copy text directly, simple advertisements, or relatively incomprehensible text. Thus, there was significant normalization for the data in order to make it more practical for machine learning and NLP purposes.
First, all references to other users and tags were stripped from the tweets. There is potentially useful information in how these references are used. For example, an analysis could create a social network and infer hate speech based on which users were tagged. These methods are beyond the scope of this project. Additionally, we performed some normalization of text with repeated letters. Namely, any sequence of more than three consecutive letters of the same type was truncated to be three letters. We made the assumption that there is a meaningful difference between ”foo” and ”fooooooooooo,” but a less meaningful definition between ”barrrrr” and ”barrrrrrr,” as this kind of repetition usually includes an arbitrary number of spurious characters.
There were additional challenges that arose from the non-traditional character set within the tweet. Emoji are used to indicate a wide array of feelings, from sadness to sarcasm. Moreover, there are significant differences in the appearance of emoji that vary between platforms. However, this project found that there was not a significant difference if emoji were included (and treated as their respective unicode characters).
The dataset was processed through a tokenization process. This broke down the text of the tweet into a series of meaningful elements. For example, individuals words and punctuation marks were treated as tokens. Many extraneous tokens, such as extra spaces were removed during this stage of the process. The goal of this data processing section was to normalize the data such that a learning model could focus on the most meaningful elements of the text and to limit the amount of noise in the predictions.
The tokenized data were then split into testing and training datasets to be applied to NLP algorithms and other machine learning techniques. The feature extraction pipeline can be seen in Figure 1,

adapted from work by Laurent Luce [5].
We performed feature extraction using a bag of words approach. We considered an information gain algorithm, but found that it did not yield results that were as reliable as the bag-of-words approach. This approach looked at how individual terms contributed to the likelihood that a particular tweet would be classified in a certain way. A similar approach has been used in spam filtering problems. This approach allows certain words, such as those that are considered to be slurs, to indicate certain labels with a high probability. Thus, they were a reasonable choice for this classification pipeline.
For each tweet, there were around 90 participants who rated the tweet. The dataset gave a confidence value that describes how much participants agreed on a certain label. Across the entire dataset, people agreed 0.86 of the time with a standard deviation of 0.17. For just the hateful tweets, there was 0.75 confidence and 0.17 deviation. Similarly, offensive tweets had 0.81 confidence and a standard deviation of 0.18 and the control were had confidence of 0.95 and deviation of 0.13.

Figure 1: Tweet Extraction Pipeline



The assessment and selection of machine learning algorithms will be discussed in Section 4.

4 Machine Learning Framework

There were two separate runs for these data. This first set performed a test in a limited domain, which offensive language in tweets by comparing the nonoffensive tweets to the set of offensive and hateful tweets. The second set of tests then attempted to distinguish all three sets of tweets from one another.

4.1 Naive Bayes

The first approach taken used a Naive Bayes algorithm. Naive Bayes constructs class labels in classifier problems by looking at interrelations. The algorithm leverages Bayes' theorem and conditional probabilities. For each tweet T, we look for the class label that maximizes the a posteriori probabilities. This dataset involves only three potential labels in a complex dataset. Many of the conventional methods for improving Naive Bayes performance, like the inclusion of tf-idf scores, were not as effective for this implementation because each individual document had comparatively few tokens. Additionally, each tweet had a maximum length of 70 tokens (though no individual tweet was this length) and the number of tokens in each tweet was roughly a gaussian distribution. Thus, the raw frequency of each term was considered, irrespective of the length of the tweet.

4.2 MaxEnt Classifier

The conditional exponential classifier is discriminative model that makes it easy to incorporate many language independent NLP models. Each data token gives a unique string that is added to a bag or words to define the model parameters. As the model is training, some features may be added such that conditional likelihoods can be computed. We then perform some optimization function to find the optimal feature weights that lead to the classification. The conditional exponential classifier is a form of a MaxEnt classifier. However, while a Naive Bayes Classifier allows each feature to affect the model independently, the MaxEnt classifier uses optimization to find weights that maximize the training data. The performance of the MaxEnt classifier relative to the Naive Bayes classifier can offer insight about how well the features of the dataset are connected to one another.

4.3 SVM

A SVM model finds a linear classifier that separates data in some high dimensional space. SVMs have extensively been used in NLP contexts, and are especially effective when there is relatively little training data. SVMs can encode high levels of complexity by using higher dimensional partitions. However, given the complexity of the data, linear models will likely be effective, as naive independence is assumed. SVMs are very effective at creating separate classifiers even when there are many features, which lead to the model being an effective choice for hate speech detection.

4.4 Other Considered Models

Neural networks were not considered because the dataset had a very sparse vocabulary, which did not lend it to inference with a neural network. With that in mind, this kind of sparse, bag-of-words approach best lends itself toward Maximum Entropy Classifiers, linearized SVM, and Naive Bayes.

5 Results

In formulating the results, we are more directly interested in how the machine learning methods are able to accurately classify the data. However, there is a significant difference between the categorization of hateful and offensive tweets, which is the an interesting effect in determining the suitability of the machine learning methods. The accuracy and classification rate of the algorithms can be found in table 1. We tested these using the machine learning algorithms in NLTK and SK-Learn (the former referencing the latter directly). We also tested algorithms found in the open source NLP package TextBlob, but found the algorithms to be too memory intensive to run on local hardware.

Table 1: Distinguishing Offensive Tweet from Control Tweet

Method	Offensive Language Accuracy	Offensive Language F1-Score
Linear SVM	0.85	0.83
Naive Bayes	0.88	0.87
Max Entropy	0.78	0.81

Table 2: Distinguishing Offensive from Hateful Tweets

Method	Offensive vs. Hateful Accuracy	Offensive vs. Hateful F1-Score
Linear SVM	0.58	0.58
Naive Bayes	0.60	0.58
Max Entropy	0.55	0.56

5.1 Discussion

Given the variance in how individuals label these text samples, we cannot expect to have extremely accurate results. All of the classifiers performed less effectively when they were classifying the higher dimensional data. This is to be expected. However, the decrease is not uniform. The Naive Bayes algorithm was best able to process the complexity of the higher dimensional space. The

Table 3: Distinguishing Hateful, Offensive, and Control Tweets

Method	Full Classification Accuracy	Full Classification F1-Score
Linear SVM	0.63	0.63
Naive Bayes	0.67	0.63
Max Entropy	0.58	0.60

Maximum Entropy classifier did not have a large drop-off in its effectiveness, but it also was not as strong to begin with.
This classifier still struggles greatly with differentiating between hate speech and speech that is only considered to be offensive. There is significant error inherent in this process, as people only agree on these tweets with a confidence of 0.75. If we were to assume random chance, then the classifier would get roughly 50% of the predictions correct. The Naive Bayes classifier with feature selection is not able to classify text nearly as well as a crowd-sourced group of people, but is still somewhat successful at differentiating the two groups. However, the algorithms are very effective at identifying text with offensive language. This is an easier task, as the probability that certain words (For example swears or racial slurs) contribute to being in the ”does not contain offensive language” bin is extremely low.
There are significant limitations to the robustness of this algorithm. However, it can still be used to identify hateful texts and to flag offensive tweets that may cross the line into becoming hate speech. The most effective algorithms were those that capitalized on independent features and when there are similar dependencies between the involved features. Given the interrelation of the data, linear classifiers were most effective, and also the easiest to run on local hardware. The Maximum Entropy classifier did not assume that features were independent, and was thus much more cumbersome and likely led to significant bias and overfitting of results. There were not marked difference in the functionality between differentiating offensive and hateful text, but the Maximum Entropy classifier was less effective at differentiating the other types of data.
If we look at the overall task, then this system is fairly good at distinguishing all types of tweets from one another. Given this training dataset, a crowdsourced group would agree with confidence of about 84%. If we define this to be an upper bound for our accuracy, then the Naive Bayes classifier is fairly competent at distinguishing these different types of text.

6 Future Work

This project could be expanded by diversifying the scope of the classifiers. For example, a more effective classifier may be one that separately checks for occurrences of hate speech related to gender, race, and other common forms of hate speech. Additionally, a more diverse dataset would allow the project to be generalizable to the hatebase dataset, and could serve as a hate speech watchdog that automates some of hatespeech detection. That said, the labeling method, and thus the efficacy of the entire system, could potentially be improved if some more rigorous definition for hate speech were used. Some legal systems use well-defined hate speech definitions for the purposes of standardizing prosecution, but there is still necessary guesswork that goes into the classification of hate speech. This will naturally limit the effectiveness of these algorithms. We have no desire to make a big-brother-esque policing system for hate speech, so we will tolerate the imprecision.
Technically, this project could additionally be improved by developing something like a wordnet for hate speech. Given that hate speech is subject to much of the nuance that complicates similar sarcasm detectors, it is difficult to differentiate hateful from offensive tweets. This would better associate the complicated relationship between certain words, like how some racial slurs are used in contexts where the are either being defined or used maliciously. With a large dataset that connects these features, it would be easier to perform domain-specific inference that would improve these algorithms.
Additionally, this project is expected to perform worse, as there is not a very rich or well-labeled dataset. For example, the Stanford NLP project uses a system where individual parts of sentences are labeled, which leads to machine learning methods being able to detect what phrases or clauses cause positive or negative sentiments. A similar system, where certain phrases are tagged as an inflection point for hate speech, would improve the functionality of this system.

7 Conclusion

From this project, we have found that it is possible to formulate machine learning algorithms that can classify hate speech roughly in accord to an averaged set of reviewers. There is a significant amount of noise between the structure of tweets and the variance of user ratings, but machine learning methods are able to accurately classify these data. There is still significant work to be done in differentiating those tweets that use hateful language but are not offensive from those that are truly hateful tweets.

Honor Code

I pledge my honor that this report has been written in accord with University Rules and Regulations, including the Honor Code.

References

[1] P. B. Gerstenfeld, D. R. Grant, and C.-P. Chiang, "Hate online: A content analysis of extremist internet sites," *Analyses of social issues and public policy*, vol. 3, no. 1, pp. 29–44, 2003.

[2] N. D. Gitari, Z. Zuping, H. Damien, and J. Long, "A lexicon-based approach for hate speech detection," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 10, no. 4, pp. 215–230, 2015.

[3] E. Kouloumpis, T. Wilson, and J. D. Moore, "Twitter sentiment analysis: The good the bad and the omg!" *Icwsm*, vol. 11, pp. 538–541, 2011.

[4] B. Liu, "Sentiment analysis and opinion mining," *Synthesis lectures on human language technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[5] L. Luce, "Twitter sentiment analysis."

[6] S. OBrien, "Twitter crackdown on hate speech backfires," May 2016.

[7] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and trends in information retrieval*, vol. 2, no. 1-2, pp. 1–135, 2008.

[8] H. Saif, Y. He, and H. Alani, "Alleviating data sparsity for twitter sentiment analysis." CEUR Workshop Proceedings (CEUR-WS. org), 2012.

[9] W. A. Schabas, "Hate speech in rwanda: The road to genocide," *McGill LJ*, vol. 46, p. 141, 2000.

[10] W. Warner and J. Hirschberg, "Detecting hate speech on the world wide web," in *Proceedings of the Second Workshop on Language in Social Media*. Association for Computational Linguistics, 2012, pp. 19–26.