Offensive Language Detection on Twitter - A Comparative Experimental Approach

Shubham Agarwal[1] Saarland University

Param Uttarwar[2] Saarland University

Saumya Agarwal[3] Saarland University

Abstract

In this course project, we focus to address the issues that exist in performing natural language processing operation on short texts and further classify them into desired categories. We work particularly on the tweets in this project and try to understand the challenges and limitations faced when classifying language based on its surface features and usage on the social platform. We implement different classification strategies – both traditional ML as well as Neural Network approach to understand their performance on such texts. By implementing different approaches and added preprocessing step, we observe that it is difficult for classifiers to model the usage of words in tweets and they perform better yet insignificantly than Naïve Bayes approach, in general.

1 Introduction

Short-length texts, such as tweets, search snippets, RSS feeds, and messages, are the most widely generated text content in current times on Internet as they have become an integral part for sharing information and communication purposes for a large share of the audience. With ever-increasing usage and generation of these unedited online content on different platforms, they also pose unexpected major issues for the users at times. Unfiltered content in tweets and Facebook posts like racial, religious comments, hate speech, offensive language, etc. which are not only offensive but inappropriate for online users are unwanted, demeaning and offensive to users. This content creates unrest and unruly behaviour among people. This may lead to cyberbullying, racial discrimination acts, inhumane activities and on a larger scale even brings about unrest on a national scale [1], [2]. Therefore, designing and training a classification model that deals with raw text as well as recognize offensive entities and smartly filter inappropriate keywords among them from the web has become a need of the hour.

Thus far, the research community has contributed significantly in the field of NLP, analyzing text data gathered from posts, messages, articles and various other sources of literature. However, due to recent surge in generation of short-text data from the social media platforms, much of the work among them only focus on the preprocessed data which is edited and the model does not attribute towards special linguistic features such emojis, abbreviations and slangs which are frequently used in casual communication. As mentioned by Zeng et al, in [3], it has been reported that lack of modelling short-texts accurately arise due to multiple reasons. Since the texts are short, the linguistic surface features, that may be helpful for classification, are missing. More importantly, the data gathered are usually sparse and there is no clear distinction between the usage of features with reference to the dataset.

In this project work, we implement two different models of classification upon short-texts to classify whether or not they are offensive. We also try to understand and confirm the limitations and challenges faced by the classifier to perform accurately on the tweets that are available to us. To summarize, the key contributions in this project work are as follows:

- We discuss the problem statement, data and feature selection in Section 2.

- Then, we discuss different algorithms which are implemented, the expected baseline accuracy, and their parameters in section 3.
- In Section 4, we report the performance of the modelled classifiers and then compare them.
- Based on the performance of our feature-based classifier, we report an error analysis on the mispredicted set of tweets with appropriate examples in Section 5.
- We conclude by presenting our views on the experiment in Section 6.

2 TWEET CLASSIFICATION TASK

2.1 Problem Statement

Tweets are written in an informal conversational style with no check on grammar or language, special characters, phrases, slangs, etc. Existing preprocessing tools do not take care of Unicode characters or emojis. Thus, they are either unclassified or misclassified in the text. We understand that they are important in terms of linguistic and semantic characteristics for classification of tweets. Therefore, our objective in this work is to come up with a generic classification model with adequate pre-processing tools which would serve as online grammar police.

2.2 Pre-processing

Pre-Processing performed on the dataset is to maintain uniformity between tweets. The model is then trained on the processed dataset. We read the **.tsv* files and extract the labels and tweets individually. In our work, we use the *nltk* modules for sentence tokenization and preprocessing steps in addition to the required tokenizer - *twokenize.py.* We added the extra preprocessing step required to split the emojis in the *twokenize.py* itself using *emoji* module. The added step would make sure that these special Unicode characters are individually classified by the model along with the words accompanied in the tweet for better results.

We further used TfidVectorizer in this task where we consider character n-grams of length from 3 to 6. Also, we used other common preprocessing techniques such as *simpletokenize* and *stopwords* to filter out irrelevant and unwanted content from data.

2.3 Feature Selection

In order to capture specific linguistic features to distinguish and classify the tweets, we extracted a few surface-based features, namely the no. of words in a tweet i.e., length and the mean word length per sentence in tweets. To our understanding, we suspected that the length of the tweet could be significantly different between offensive and non-offensive ones. For example, the non-offensive tweets, in general, are longer and does contain many special characters when compared to the offensive text.

Similarly, the mean word length of offensive texts is suspected to be lower than the non-offensive texts as mentioned in [4]. We trained our classification model based on these two features and implement the algorithms discussed further. Other possible surface feature could be the use of special characters per sentence or words as well as the Unicode characters in a tweet.

3 CLASSIFICATION MODELS

3.1 Naïve Bayes Classifier

Naïve Bayes classifier for NLP tasks has a limited capacity with no functionality to handle context/history. It trains over the given entity without associating it to the previous entity. We understand that it has little practical significance as this model is not a feasible solution as tweets usually follow a topic and context. These entities in tweets individually do not hold a meaning or may not sound offensive if seen out of context. For remembering a context, there is a need to come up with a better solution than a Naïve Bayes classifier. We implemented our surface feature-based Naïve Bayes Classifier as a baseline classifier following the works in [5]. We further compared the baseline accuracy obtained from this model with other models that we implement further. This methodology helped us to find the best-suited classification model for the tweet-like text processing tasks.

3.2 SVM Classifier

After setting up the baseline classification model, we adopted the traditional machine learning approach of classification for this task and develop an SVM model to classify tweets. We implemented both the linear and RBF kernel in SVM to understand the distance of decision boundaries to the data points in case of tweets as explained in [6]. We observed the performance of the classifier by manually altering the values for hyperparameter and gamma for this task instead of performing the search.

3.3 Neural Network Approach

After we observed the performance of the SVM classifier model on the given tweet dataset, we further implemented a neural network to classify the dataset into categories. We constructed a simple neural network as demonstrated in [7] along with the same features that were initially extracted. We additionally included PCA with fewer features as we noted from the behaviour of the other two models and inferred that the dataset has sparse features which makes it tough to classify accurately. The neural network consists of 1 input layer along with 2 hidden layers.

Apart from PCA, we also performed under-sampling of dataset since we observed that the dataset contained tweets in the ratio of 70:30 as per the categories. For this model, we selected the rows in the ratio of 50:50 for training in each batch where the batch size of 64 gave the best result. Also, the maximum number of epochs that gave the best result was 100 as increasing the no. of epoch resulted in overfitting and thus, less accuracy.

4 RESULTS

We ran each of the classification models on the same dataset and with same pre-processing steps. The overall accuracy score which we recorded for each of the above-mentioned models are as follows:

Classification	**Accuracy**	
Models	**dev.tsv**	**Test.tsv**
Naïve Bayes Classifier (Baseline)	70.4%	47.0%
SVM Classifier – Linear Kernel	66.1%	48.1%
SVM Classifier – RBF Kernel	63.9	47.9
Neural Network	**71.9%**	**71.9%**

Table 1 Classification Accuracy of different classifiers

The confusion matrix for the above implementation is also available in the code output.

As highlighted in Table 1, we observe that the Neural Network best performs to decide whether or not tweets are offensive when compared to other classifiers. Since we set our baseline to the value obtained from the Naïve Bayes Classifier, we observe that even though Neural Network gives better results, yet they are insignificantly greater than that from the former approach. We understand that since Naïve Bayes does not model the history associated with the tokens, it fails to yield highly accurate results. While the neural network has the capability to capture context, we suspect that due to limitation of choice in our model, i.e., no. of hidden layers, etc., maybe the reason behind the insignificant result.

When observing at the outputs obtained from the SVM classifier, we infer that margin-based classifiers perform poorly on these tasks and are not among best of choice for better results. Thus, we infer that an alternate neural network approach which is able to model short and long-term dependencies within the text may be a better choice for this task. Implementation of LSTM or Gated Recurrent Units in RNN may be the potential next step to evaluate the performance on this dataset.

5 ERROR ANALYSIS

After finishing up with the implementation and execution of our surface features-based classifiers, we selected the best performing model and analysed the mispredicted labels when testing against *dev.tsv*. We observed that there are in total of 281 mispredictions. We manually analysed them and tried to understand the reason behind its misclassification. We now discuss potential issues in the classification models with respect to the gathered data.

Majority of the tweets which are classified as offensive while they actually not seem to be the victim of lack of context while classification as merely the existence of negative words such as *pissed*, *drug* or *shit* leads to the classification of whole tweet as offensive. For example:

```
Predicted: OFF, Actual: NOT, Tweet: @USER Comey has pissed on the us
Constitution..he is a rat and needs to go away or face consequences

Predicted: OFF, Actual: NOT, Tweet: @USER @USER @USER I wonder if she
is still in love with the drug dealing boyfriend that worked at Burger
King.
```

Other tweets which are classified as not offensive while they actually are, we observe that the misclassification occurs due to the fact that the offensive words present in the tweet are either not recognized, misspelt or abbreviated other than the lack of context issue which makes it useless for the classifier. For example:

```
Predicted: NOT, Actual: OFF, Tweet: @USER Hogg is such a dolt! In a
mass shooting it's rarely a long range shooting incident. It's usually
close range inside a building of area thats a Gun Free Zone. Like
shooting fish in a barrel. Snipers are rare in mass shooting events.
D.C. and Vegas were exceptions.

Predicted: NOT, Actual: OFF, Tweet: @USER said on @USER he doesn’t
think we should be selling AR15’s but Liberals deny it and say he is
pro gun! The AR15 is not an assault weapon it is a sporting rifle.
#KeepTexasRed vote @USER
```

There are certain tweets in the dataset which seem to be labelled otherwise or are inconsistent. For instance, the tweet with offensive words which are labelled as not offensive or vice versa destroys the basis for classification and thus the sparsity increases. For example:

```
Predicted: OFF, Actual: NOT, Tweet: @USER @USER Fuuckkkk youuuuu

Predicted: NOT, Actual: OFF, Tweet: @USER Good! #Antifa is violent
fascism.
```

While some tweets which are originally labelled as offensive or not offensive, some, in general, doesn’t really do justice with their labels. This also leads to the misprediction and we assume that the predicted label is actually correct by the classifier, though not reflected in the accuracy results. For example:

```
Predicted: OFF, Actual: NOT, Tweet: @USER @USER Miriam

Predicted: NOT, Actual: OFF, Tweet: @USER You are my hero
```

Moreover, there are some tweets for which it is difficult to predict what may have been the reason for otherwise prediction. For example:

```
Predicted: NOT, Actual: OFF, Tweet: @USER You are my hero

Predicted: OFF, Actual: NOT, Tweet: @USER that's unepic
```

Though there may be many other categories of mispredicted tweets, we limit our error analysis until this point. The file containing the mismatched tweets and their labels as above are included in the submission file for further analysis.

6 CONCLUSION

In this course project, we focussed to understand the requirements necessary to model short-texts, i.e., tweets, provided to us from the Twitter dataset and classify them into binary categories based on textual and linguistic features. We implemented three classification models in total, where the accuracy obtained from Naïve Bayes was used as a baseline to compare against the output of the other two models. Additionally, we also observed the behaviour of the model along with added pre-processing step. We compared the results obtained from all the classifiers and discussed our observations with respect to preprocessing steps and the respective training models. In the end, we analyze errors in prediction by the classifier to understand the root cause of such misprediction based on the gathered mismatched labels.

We infer from this project work that short texts like tweets are comparatively hard to classify than normal text classification tasks due to limited or no surface features in the text. Also, the classification model requires the context of the word being processed in the tweet to associate with either of the labels as just the mere existence of any word in the tweet shouldn't decide on its category, in an ideal sense. We believe that approach based on Recurrent Neural Networks may yield significantly better results as the network would be able to capture long-range dependencies and thus contexts, unlike the models implemented in this work.

Reference

[1] Gaydhani, Aditya, Vikrant Doma, Shrikant Kendre, and Laxmi Bhagwat. "Detecting hate speech and offensive language on twitter using machine learning: An n-gram and tfidf based approach." *arXiv preprint arXiv:1809.08651* (2018).

[2] Watanabe, Hajime, Mondher Bouazizi, and Tomoaki Ohtsuki. "Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection." *IEEE Access* 6 (2018): 13825-13835.

[3] Zeng, Jichuan, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King. "Topic memory networks for short text classification." arXiv preprint arXiv:1809.03664 (2018).

[4] Prusa, Joseph D., Taghi M. Khoshgoftaar, and David J. Dittman. "Impact of feature selection techniques for tweet sentiment classification." In *The Twenty-Eighth International Flairs Conference*. 2015.

[5] Xu, Shuo. "Bayesian Naïve Bayes classifiers to text classification." *Journal of Information Science* 44, no. 1 (2018): 48-59.

[6] Joachims, Thorsten. "Text categorization with support vector machines: Learning with many relevant features." In *European conference on machine learning*, pp. 137-142. Springer, Berlin, Heidelberg, 1998.

[7] Prasanna, P. L., and D. R. Rao. "Text classification using artificial neural networks." *International Journal of Engineering & Technology* 7, no. 1.1 (2018): 603-606.