Topic classification of Tweets

MANOJ JOSHI, Indiana University
SHRADHA BARANWAL, Indiana University

Twitter is one of the most used social media platforms of this century. It is a wide-spread platform not belonging to a particular sect of people making it a source for all kinds of information from all fields of expertise. We believe that such a powerhouse of information can create a lot of confusion in terms of understanding the context of the tweet in an ocean of them. There is ample of literature present today for the classification of tweets in different categories which can help the users gain insight into the context of the same. However, after going through some of the research papers it is evident there is a dearth of experimentation with latest algorithms like XGBoost and Neural Nets. We have tried to explore these algorithms to classify the tweets in categories along with comparison with the more widely used algorithms like Logistic Regression and Naive Bayes.
We have classified the tweets into four basic categories using the text from the tweets. The four categories are Entertainment, Sports, Politics and Technology. We have experimented with only "bag of words" approach and compared the results of the classification between all the algorithms using accuracy as the measure. We have also compared the accuracy across the categories to understand if categories played any specific role with any particular algorithm.

Additional Key Words and Phrases: Topic classification, Twitter, Niave Bayes, Logistic, Random forest

ACM Reference Format:
Manoj Joshi and Shradha Baranwal. 2018. Topic classification of Tweets. 1, 1, Article 1 (April 2018), 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Twitter is the most popular micro-blogging website which is real time in information dispersion. One of the most unique features of tweets is the limit on the number of characters of word of each tweet. Today, 500 million tweets are generated per day. The numbers only indicate how much information can be generated in a single day. This is one of the most challenging aspects of using the website. Even though the users follow only the accounts they are mostly interested in(have a context about), a trending topic or a tweet which is not contextual to the user very frequently appears on his/her feed. We feel that such if a model is generated to guess the category of the tweet, it can help user get a gist of the information in the tweet without a lot of hassle. To tackle the problem, we have classified the tweets in four basic categories of Entertainment, Sports, Politics and Technology. We have tried to learn the patterns in the tweet text specific to the category by using the Bag-of-words approach and then experimenting with numerous classification models. This makes our problem a classic multi-class classification problem and we have used and compared the results from some classic algorithms like logistic regression, Naive Bayes, Random Forest along with some experimental new algorithms like XGBoost and Neural Nets

2 LITERATURE REVIEW

This section presents the previous work done in this research area and a brief description of their published work. Kathy et. al[1] have classified the trending topics into various broad categories. They have considered 18 categories for the task which makes the classification challenging. "We find that trend names are not indicative

Authors' addresses: Manoj Joshi, Indiana University, manjoshi@iu.edu; Shradha Baranwal, Indiana University, sbaranwa@iu.edu.

of the information being transmitted or discussed either due to obfuscated names or due to regional or domain contexts."[p 252]. They have used two major approaches to achieve the results. One of them is by using Bag-of-words technique where in the keywords from each tweets of specific category are used to "define" that category. They have experimented with models like Naive-Bayes, SVM, Logistic Regression and K-NN algorithms to name a few. Along with the above technique, they have also used network information by identifying the biggest "influencer" for a particular category. The "influence" of a twitter account was calculated using the influence of his/her followers. The culmination of both the approaches has helped them with good experimental results as mentioned by them in the paper.[1]
Sriram et al.[2] in his research paper has talked about a novel approach of classification of tweets in 5 categories which are News, Opinions, Events, Deals and Personal messages. For this classification, they have emphasized on the "authorship" of a tweet rather than the tweet itself. "Authors generally adhere to a specific tweeting pattern i.e., a majority of tweets from the same author tend to be within a limited set of categories."[pg 2]. The authors have claimed that the proposed novel approach works better than the traditional Bag of words approach. They have also managed to find the emphasis of words in each tweet by using their characteristic like either capital letters or repetition of letters(eg veeeery).
In a short paper by Daniel Schultz and Sundeep Jolly[3], the authors have identified a well known issue of incorrect tweets getting tagged to wrong hashtags as well as missing the hashtags. This leads to missing of tweets when researching them on based of hashtags. Using a similar method of bag-of-words approach and modeling algorithms like Naive Bayes, K-NN as well as SVM for topic classification, the authors have tried to tackle the issue by suggesting a solution of automatic tagging of tweets with the help of these classifiers.
P. Selvaperumal et. al[4] have discussed a similar way of tweet classification. The authors have neatly explained the procedure of data collection, pre processing and classification. A novel introduction in their data processing approach is the handling of URLs. The authors have collected the data from the website of the URLs. This adds an important layer of information while training the model.

3 DATA

The tweets required for the classification task were obtained using Tweepy, a Python library for accessing the Twitter API[8].

3.1 Data Collection

As part of the data collection phase, 5 hashtags were included for each of the categories. For instance, tweets with hashtag "#congress" were obtained and those tweets were categorized into Politics. The hashtags used to obtain tweets for each of categories are described below:

- Politics - "#congress", "#modi", "#brexit", "#senate", "#NRA"
- Sports - "#superbowl","#nba", "#cricket", "#olympics", "#ManU"
- Technology - "#virtualreality", "#blockchain", "#cryptocurrency", "#IoT" , "#cloudcomputing"
- Entertainment - "#musicfestival", "#oscars", "#InfinityWar", "#hollywood", "#bollywood"

3.2 Data Pre-processing

After gathering tweets for each of the categories, we noticed that there were many re-tweets. To have a good quantity of unique tweets in our data set, we removed all the re-tweets.

Also, as part of data cleaning phase, we noticed that a lot of tweets had many new line characters in them. These new lines did not make any sense towards the classification and hence we removed the additional new lines for each tweet. Using the same idea, since usernames did not give a lot of information for the classification task, we removed any word that began with the character "@".

Lastly, we did not want the classification models to give a lot of weightage to the hashtags themselves and wanted the models to classify tweets solely based on the text of the tweet. For instance, there can be a lot of "Politics" tweets with the hashtag "#politics" which would cause the model to give less importance to the text and more to the hashtags making the model more biased. For this reason, we remove all the hashtags from each tweet and retain only the text.

After the data collection and the pre-processing steps, we were left with - 2733 tweets for Politics, 2459 tweets for entertainment, 2726 tweets for Technology and 2490 tweets for Sports. The number of tweets for each class were nearly the same and this would naturally avoid the classical problem of class imbalance for classification. The complete data was stored as a tuple (Tweet, Category) in a csv file. The reason for storing the data as a csv file instead of a widely used approach like a pickle is that we often wanted to have a look at the data and see few random tweets and their categories. Using a pickle this would be tedious since we would have to un-pickle it and index it randomly.

4 ALGORITHM AND METHODOLOGY

4.1 Vectorization

tf-idf : This acronym stands for term frequency-inverse document frequency. It is a way of measuring the weights of the words not just in terms of the frequency of occurrence but also in terms of the number of documents containing the word. The weights are calculated such that a word with most frequency in the lowest number of documents has higher weights. This implies that stop words which are frequent in all the documents have low weights. Similarly, words with very low frequency in documents have low weights.
TfidfVectorizer :This is a python function from sklearn library used to create a matrix of weights as described above. The function creates a matrix of every words as a feature (barring the stop words and the words with a frequency lower than the threshold given as input) with weights for each document. The output of this function acts as our independent variable in the model.

4.2 Logistic Regression

Logistic regression is one of the classic algorithms used for binary classification of datasets using the sigmoid function. The sigmoid function helps in predicting the probabilities of a particular data row belonging to a class. The class with higher probability wins.
However, in our case, the question requires a multi-class classification due to presence of four categories in dependent variable. In such case, the *LogisticRegression* model from sklearn uses the concept of one-Vs-other. This technique entails classifying the data into binary classes for every class as one versus other classes. This helps in getting probabilities for every class and the class with highest probability wins.
The accuracy obtained with logistic regression was the best among all the other methodologies.The cross validation accuracy was 63.83%

4.3 Naive Bayes

Naive Bayes is a probabilistic classification based on the Baye's theorem and assumes that the features are independent to each other. It is widely used in the document classification task based on the word frequencies of the documents as features. This algorithm goes through the entire dataset once and build a probability table. For every new sample seen, the classifier calculates the posterior probability using the probability table constructed in the training phase. Unlike other classification algorithms, this is not an iterative algorithm.

4.4 Random Forest

In Random forests, multiple random decision trees are created using only a small random subset of the features. Samples of the dataset are taken with replacement, but the trees are constructed in a way that reduces the correlation between individual classifiers. Specifically, rather than greedily choosing the best split point in the construction of the tree, only a random subset of features are considered for each split. Using Random Forests also removes the necessity of pruning trees which would otherwise be required to prevent over-fitting of the model to the training dataset. We tuned the hyper parameter "n_estimators" which indicates the number of trees constructed. Keeping n_estimators=70 gave us the best accuracy.

4.5 XGBoost

"XGBoost is an acronym for eXtreme Gradient Boosting. It is loosely based on the parallel execution using decision trees and aggregation of their results"[2]. We have used "multi:softmax" as the objective function for the current problem. This objective function is used for multi-class classification and outputs the class as output. One of the challenges while using XGBoost was deciding parameter and hype-parameter. We referred the template mentioned by Owen Zang in his lecture on "How to win Kaggle competitions". Below are some of the tree based parameters(booster = "xgbtree"):

(1) n_estimators: This is the number of passes over the dataset using the previous passesâĂŹ misclassification data. This is basically the number of tree that is created during the process of training. By default the value is set to 100.
(2) max depth: This parameter is used to avoid overfitting in model training. Greater the value of max depth, greater will be the accuracy for training data and more will be overfitting. For the purpose of our problem, we have set the value of this parameter to 5
(3) learning_rate: This parameter is the learning rate used during the process of gradient descent. The value we used for the problem was 0.01.
(4) colsample by tree: This specifies the fraction of columns to be used for splitting while creation of trees. The value for this parameter was 0.4.
(5) subsample: This specifies the fraction of total rows to be considered for every decision tree. The value for this parameter was 0.6.
(6) min child weight: This refers to the minimum number of instances possible in at a leaf node. In case a leaf node has reached the min child weight limit, no further partitioning is done. The value of this parameter 20 as the dataset is huge and consists around 10000 datapoints.

The accuracy obtained from XGBoost was rather disappointing. We believe that this might be because of no tuning of hyper-parameters. We believe that using Grid search method to hyper tune the parameters and using them might increase the model performance.

4.6 Neural Networks

The recent advancements in the application of Neural Networks in the task of Natural Language Processing motivated us to use the state of the art neural networks like Convolutional Neural Networks (Add reference to CNN) and Recurrent Neural Networks(Add reference to RNN).

Neural Networks work on numerical data and we had to vectorize the text to feed into both the neural networks. The first of text vectorization involves creation of a matrix of sequences for each tweet whose dimension is (N*D) where N is the sequence length and D is the embedding dimension of each word in the sequence.

Even though each tweet can be of different length, we had to keep the dimension same throughout. To achieve this, we used the Keras Tokenizer (a text processing module in Keras) to convert the tweet to a sequence of numbers. If the length of the sequence was less than N, we append zeros at the beginning indicating that there was no word there. To represent each word as an embedding vector, we used the Stanford Glove Embeddings (put reference) trained on the Wikipedia corpus. This embedding contains 100 dimensional vector for each word. Using this embedding for our tweets makes every tweet as a matrix of dimension N*100. We kept this as our input vector and experimented with Convolutional Neural Network and Recurrent Neural Network.

To train a neural network, we divided the data into two parts, 80% of the data as training data and 20% of data as the test data. We evaluate our neural networks on the test data to see as to how they generalize across various classes.

4.6.1 Convolutional Neural Network. : Convolutional Neural Networks have been very successful in computer vision tasks such as Image Recognition, Object Detection, Emotion Detection etc. A variant of this called the 1D Convolutional Neural Network has been applied to various NLP tasks. At a higher level, convolution is a mathematical operation done on a matrix to capture the important features and reduce the dimension of the overall matrix as repeated convolution operations are applied. In terms of language, 1D convolution captures the n-gram features of the text i.e., the n-grams of the tweet are represented as a feature for each 1D convolution operation. Here, the 'n' depends on the size of the convolution filter i.e., if we use a size of 5, it represents 5-grams of the tweet. Our 1D Convolutional Neural Network (CNN) had a single 1D CNN layer with 128 filters each having a size of 5 (5-grams). This CNN layer was followed by a Max-Pooling layer. The intuition of using a Max-Pooling layer is to extract the most relevant 5-grams from a set of 5-grams. In our network, we set the size of Max-Pooling to 5. This extracts the most important 5-gram required for classification from the set of 5-grams.

4.6.2 Recurrent Neural Network. : Recurrent Neural Networks(RNN) have been widely used in complex NLP applications such as Machine Translation, Author Identification, Chatbots etc. These type of neural networks give importance to word order and process sentences sequentially. A variant of RNN's called Long Short Term Memory (LSTM) have been used in practice since RNN's suffer from a well known problem called vanishing gradients which essentially means that the neural network will not be able to learn anything after some point in time.

In our network, we used an extended LSTM network called the Bi-Directional LSTM which sees information from the future and from the past before processing the current word in the sequence. Our network architecture had a single Bi-Directional LSTM followed by 3 fully connected layers. These fully connected layers are used to capture the interactions between the features extracted by the LSTM layers. After few iterations of training, we realized that the network was over-fitting very quickly. Hence, we added few dropouts between every two layers. Dropouts shut down random neurons during training enabling other neurons in the layer to learn accordingly.

5 AUTHOR CONTRIBUTIONS

We initially distributed the data extraction process by assigning two categories each. The code for the process of data extraction was mainly done by Manoj. The data preprocessing of removal of hashtags and usernames was implemented by Shradha. The Vectorization of the dataset was done by Manoj followed by running Random forest, Naive Bayes models.
Shradha used the same vectorized data generated in the previous step and constructed the models Logistic Regression and XGBoost for dataset.The vectorization of the neural nets using Glove Embeddings followed by CNN, Bi-directional LSTM was done by Manoj.
The report was written by both, contributing to the sections one worked on.

6 RESULTS

We have plotted the accuracy of each algorithm we have used. The predictions were obtained using the *cross_val_predict* function from $model_selection$ package of sklearn. The predictions were then compared to find the correct predictions to calculate the accuracies.
Unfortunately, the sophisticated algorithms(Neural Nets and XGBoost) could not outperform the classic Logistic regression with an accuracy of 63.83%. The next best accuracy was given by random forest which is 55.83%.

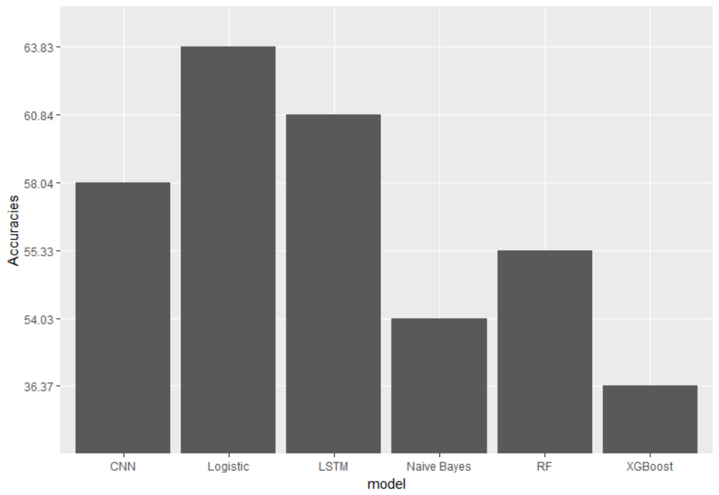

Fig. 1. Bar plot specifying the accuracies of each algorithm used

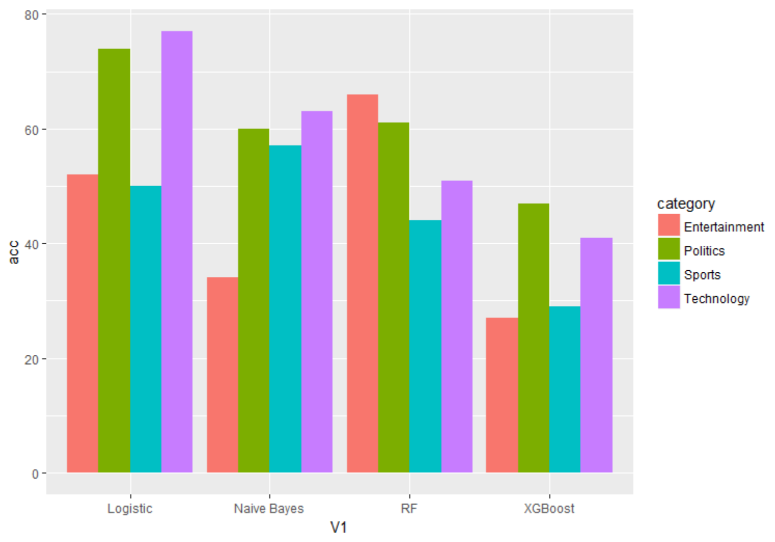

Fig. 2. Bar plot specifying the accuracies of each algorithm used

As seen in the Fig 2, we have also plotted a comparative accuracies across the category for every model we have used.

7 CONCLUSIONS

The results indicate that Logistic Regression outperforms all the algorithms. While going though some tweets manually, we saw that there were a lot of tweets with no text at all but with just a URL. These tweets can be categorized only when you go to URL and see the content. The neural networks were able to overfit the data easily and did not perform well on the test data. The reason for this is usage of free grammar in tweets - many tweets did not have a well formed sentence leading to many missing words from the embeddings which resulted in too many zero vectors. To conclude, the topic classification of tweets is a hard problem to solve with just the tweet text. A probable approach for future research would be including other features which incorporate the user information such as the topics of interest for the user based on historical tweets, user connections etc.

REFERENCES

1 Kathy Lee, Diana Palsetia, Ramanathan Narayanan, Md. Mostofa Ali Patwary, Ankit Agrawal, and Alok Choudhary, "Twitter Trending Topic Classification", in 2011 IEEE International Conference on Data Mining Workshops
2 B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu, and M. Demirbas, Short text classification in twitter to improve information filtering, in Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval, 2010, pp. 841842.
3 http://courses.media.mit.edu/2010fall/mas622j/Projects2010/SunnyJolly_DanSchultz.pdf
4 P. Selvaperumal and A. Suruliandi, "A short message classification algorithm for tweet classification," 2014 International Conference on Recent Trends in Information Technology, Chennai, 2014, pp. 1-3.
5 http://firstmonday.org/article/view/2745/2681
6 Stanford Glove Embeddings. Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014 https://nlp.stanford.edu/pubs/glove.pdf

7 https://en.wikipedia.org/wiki/Naive_Bayes_classifier
8 http://tweepy.readthedocs.io/en/v3.5.0/getting_started.html