Topic based stance detection on Twitter data

Adrija Roy adriroy@indiana.edu
Abhishek Singh aasingh@indiana.edu
Shubhankar Mitra shmitra@indiana.edu

Abstract

In this paper we are going to discuss our findings in attempting supervised classification of tweets based on their stance towards a pre-defined target using various linguistic feature sets. Using f-score metric to judge classification accuracy, we have been able to achieve accuracy of above 60%.

1 Introduction

Stance detection is the process of determining whether a given statement is positive or negative or neutral with respect to a certain subject. Since the US election just got over, we can take an example from there. Let us take the example of a single tweet and consider the target to be Donald Trump, well stance detection will help us to identify whether the given tweet is in favor of Trump or if it is against him or is neutral. In this example the target is a person, but it can be any subject; it can be an organization or a region or a religion or a global phenomenon. So stance detection can be extremely useful in taking up any text and determining automatically whether or not that text is conveying points in favor of a target or is against or whether it is neither in favor nor against the said target [1]

2 Related Work

In the semEval 2016 paper by Mohammad et al. [1] the authors take a twitter dataset and perform stance detection on it. The dataset that we are going to work with, in this study, is the same one that Mohammad et al. used for their paper. The data consists of 4870 English tweets annotated towards six targets: 'Atheism', 'Climate Change', 'Hillary Clinton', 'Abortion', 'Donald Trump' and 'Feminism' along with the stance for that target. Stance consisted of 3 categories: 'Favor', 'Against' and 'None'

3 Methodology

We divided the dataset which was used in task 1 in [1] into two sets. One consisting of 2914 tweets used for training our classifier, and 1249 for judging accuracy as the testing data. Classifiers were trained on each target separately and tested using the testing dataset of each target.

3.1 Dataset description

The dataset we used for our project is the same as the one used in [1] for task 1. So there were in total 4870 tweets in the entire dataset, out of which we used 70 percent for training and the rest for testing. A more detailed description of the dataset is given in Table 1.

TARGET	Total number of tweets	Training Tweets	Training Favor %	Training Against %	Training Neutral %	Testing Tweets	Training Favor %	Training Against %	Training Neutral %
Atheism	733	513	18%	59%	23%	220	15%	73%	13%
Climate Change	564	395	54%	4%	43%	169	73%	7%	21%
Feminism	949	664	32%	49%	19%	285	20%	64%	15%
Hillary Clinton	984	689	17%	57%	26%	295	15%	58%	26%
Abortion	933	653	19%	54%	27%	280	16%	68%	16%
Total	4163	2914				1249			

Table 1: Data description

3.2 Evaluation Metric

We have used Precision Recall metric to evaluate our models. Using precision recall, we calculate the F-score for each of the six targets in our dataset. We use F-score as the measure of accuracy and we will discuss about it a bit later. Precision-Recall is one of the most widely used evaluation metrics and can be described as follows:

Precision: Precision, also called positive predictive value, is a very useful tool when it comes to evaluate classification models. It can be defined as the ratio of the total number of true positives divided by the number of instances that are deemed positive by the model. So it is given by: $\frac{true\_positives}{true\_positives+false\_positives}$.

Recall: Recall, also called sensitivity, is also very useful and is used to identify how many of the positive classifications were actually correctly classified. So it will only take into account the records that were positive in the actual dataset and check how many of them were correctly classified. So it is given by: $\frac{true\_positives}{true\_positives+false\_negatives}$.

So for example in our dataset we are classifying tweets about Hillary Clinton to be for her or against her. So, let us assume there are 500 tweets for Hillary and 200 are against her. So if our model correctly classifies 300 of the tweets for her, and assigns 100 of the tweets to be for her incorrectly, then for our model the precision will be: 300/400 and recall will be 300/500.

F-score: Using precision and recall we calculate the average F-score for each of the six targets in the dataset. The F-score metric is very useful in measuring a model's accuracy while performing binary classification. So we can ascertain the validity of our model using the F-score. So how we calculate the average f-score of a target is as follows:
First we calculate the Precision for positive stance towards the target, let us denote that

as P_{favor} and let us also calculate the recall for positive stance towards a given target, let us represent that by $R_{favor.}$ Similarly, we can calculate $P_{against}$ and $R_{against}$ for the target. Now the F score for tweets to be in favor of the target is given as:

$$F_{favor} = \frac{2*Pfavor*Rfavor}{Pfavor+Rfavor}.$$

$$F_{against} = \frac{2*Pagainst*Ragainst}{Pagainst+Ragainst}.$$

Now once we have these we calculate the average F-score for the target by taking an average of these two values. So for the target, the f score is given by:

$$F_{target} = \frac{Ffavor+Fagainst}{2}.$$

3.3 Linguistic Feature-set descriptions

For all the feature set cases, feature set was created using training data for each target separately and the same feature set were used to create features in the test data for the same target. We attempted training our models with various feature sets individually. These feature sets are described below.

Feature set 1: To get a baseline understanding of performance, we used all words as features from the training dataset and used the same vocabulary for creating features in the test set. This excluded the punctuations and special symbols but included the hashtag words. Each word represented as a feature with its occurrence count in a tweet as its value for that tweet.

Feature set 2: We then limited the vocabulary to include only the nouns, verbs and adjectives identified in the training data. Nouns, verbs and adjectives were identified using Stanford POS tagger [2]. This excluded the hashtags and mentions. Each noun, verb and adjective in the training data was represented as a feature with its occurrence count in a tweet as its value for that tweet. Similar features were created in test data using the nouns, verbs and adjectives identified in the training data.

Feature set 3: Then we created feature sets of unigram to pentagram whose minimum frequency in the training dataset of the respective target was 5. This excluded stop-words ('i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now') but included hashtags and mentions. This is a more optimized feature set as only frequently occurring word sets are being

considered but extremely high frequency stop-words are being removed.

Feature set 4: We then introduced features from the MPQA subjectivity lexicon [5] to optimize our feature set further. We used these features in an attempt to see if subjectivity lexicon features can boost up the performance when combined with the feature 3. Four features were created: count of POS and word match for negative weaksubj, count of POS and word match for positive weaksubj, count of POS and word match for negative strongsubj and count of POS and word match for positive strongsubj.
These four features created from MPQA were then used along with our optimized feature set 3 to get feature set 4.

Feature set 5: After combining the subjectivity lexicon, we then decided to investigate the features from arguing lexicon [6]. Fifteen features were created each one representing count of matches in the tweet for the corresponding argument type (eg: assessment, doubt) as defined in the lexicon. We used these features and merged them with the optimized feature set 3 to get a new feature set 5.

Feature set 6: We then looked into MALTparser [4] and created features of dependency triples with frequency of more than 5 in the dataset. The features represented count of these high frequency dependency triples in the tweets. We decided to use these features by combining these features along with our optimized feature set 3.

Feature set 7: We then used Stanford Named entity recognizer to create features [3]. Three features of count of location, person and organization identified in each tweet were created. We then combined feature sets 3 to 6 and merged it with the NER feature set to come up with a new comprehensive feature set.

3.4 Machine learning algorithms used

TiMBL : TiMBL[7] is a software package that implements multiple memory based classifiers and learning algorithms. It is very much useful for supervised machine learning, and hence finds a lot of use in natural language processing. It employs a function approximation method, so it is logically very easy to grasp. We store each data point in memory, and hence making a prediction about an output, which will result from some input variable, is quite easy to understand. Due to the decision-tree based implementation of TiMBL, it is quite fast as well and performs better than most of the other algorithms when it comes to classification. So TiMBL is quite easy to follow and also performs quite well. We used our train data to train TiMBL for each feature set and target separately and used it to predict the stance class in the testing dataset for the corresponding feature set and target.

Random forests: Random forests [8] are also known as random decision forests. As the name suggests a random forest is created by combining multiple decision trees. So we model the training data set to come up with all the decision (classification) trees, which

forms the random forest. Now during testing, when we provide an input to get a prediction, the input is given to all the classification trees of the random forest, and we get a classification from each of them. Then random forest returns that classification as the prediction, which was predicted from the majority of the underlying classification trees.

There are many reasons for us to choose to work with random forest. It is regarded as the most accurate classifier and works well with large databases. It also can handle high dimensional data really well, and given the number of features we had, we thought it would be ideal to use random forest.

4 Results

	All words as features (Feature set 1)	*Nouns, verbs and adjectives as features (Feature set 2)*	*Optimised feature set (Feature set 3)*	*Optimised feature set with MPQA subjectivity lexicon (Feature set 4)*	*Optimised feature set with arguing lexicon (Feature set 5)*	*Optimised feature set with MALTparser feature set (Feature set 6)*	*Final comprehensive feature set (Feature set 7)*
Atheism	0.46	0.53	0.55	0.54	0.55	0.57	0.56
Climate Change	0.4	0.36	0.37	0.38	0.38	0.39	0.37
Feminist Movement	0.54	0.52	0.53	0.55	0.52	0.55	0.52
Hillary Clinton	0.47	0.42	0.53	0.47	0.54	0.53	0.52
Legalization of Abortion	0.61	0.53	0.6	0.55	0.52	0.6	0.58
All targets together	0.65	0.6	0.61	0.62	0.61	0.62	0.63

Table 2 Results for Random Forest Classifier on Dataset

	All words as features (Feature set 1)	*Nouns, verbs and adjectives as features (Feature set 2)*	*Optimised feature set (Feature set 3)*	*Optimised feature set with MPQA subjectivity lexicon (Feature set 4)*	*Optimised feature set with arguing lexicon (Feature set 5)*	*Optimised feature set with MALTparser feature set (Feature set 6)*	*Final comprehensive feature set (Feature set 7)*
Atheism	0.42	0.21	0.53	0.50	0.49	0.52	0.51
Climate Change	0.03	0.21	0.33	0.35	0.33	0.34	0.33
Feminist Movement	0.33	0.41	0.42	0.42	0.42	0.43	0.42
Hillary Clinton	0.18	0.36	0.49	0.52	0.49	0.47	0.49
Legalization of Abortion	0.26	0.44	0.43	0.46	0.42	0.43	0.43
All targets together	0.34	0.45	0.50	0.55	0.47	0.51	0.53

Table 3 Results for TIMBL classification on Dataset

5 Analysis of result

Table 2 presents our result obtained from performing results on TIMBL classifier. TIMBL classifier offers two algorithms to chose from- IB1 and IGTREE. We tried the results using both algorithms and decided to go with IGTREE since it was performing better on all the datasets compared to IB1. We used the algorithm with weighted overlap and MVDM in place with 5 and 9 nearest neighbors. The column values shown in Table 2 correspond to a target F-score values obtained by averaging F-score for "Favor" and F-score for "Against". We can observe in the table that TIMBL seems to be doing better than baseline classifier on most of the target data. The oddity in classification of Climate Change dataset can be explained by the fact that it's a highly unbalanced class. As far as the feature-set that works best across all targets, optimized feature set with MPQA subjectivity lexicon is performing pretty well across all data. It can be explained by the extra supervised weighted subjectivity information that's going in the feature vector, thus helping the classifier use better features for decision making.

Apart from TIMBL, we applied random forest classifier which further enhanced our classification performance over baseline as well as TIMBL. The Random Forest classifier already does well across all targets even with the sparse feature vector of using all the words as features. But, substantial improvements can be seen when we use the comprehensive feature vector with optimized words, subjectivity & arguing lexicon coupled with dependency parsing information. The improvements are several orders of magnitude over baseline as well as TIMBL (10-15% improvement).

6. Conclusion and Future work

We approached the problem of Stance detection towards a specific target from tweets using Machine Learning and Natural Language Processing tools. The diminutive distinction between sentiment analysis and stance detection makes the task much more challenging by relying more on extracting relevant information towards the target of interest and using them as features to further optimize out classification model. So far, we had success in creating a model that is performing better than baseline classification using POS, lexicon and parsing information modelled using memory based learning algorithms and Random Forest classifier.

We believe there's a lot more room for improvement here, given the rich set of information that is delivered through tweets including symbols, hashtags, links time of tweet, and emoticons. We look forward to extracting these metadata from textual tweets and making them as features fed to complex machine learning algorithms, potentially giving state-of-the-art performance on Stance Detection.

Acknowledgment
We are grateful to Prof. Sandra Kuebler for introducing us to this topic and providing datasets to works on.

References

[1] Sem-Eval-2016 Task 6: Detecting Stance in Tweets. Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Shobhani, Colin Cherry, Xiaodan Zhu. Proceedings of Sem-Eval 2016.
[2]Stanford NLP POS Tagger http://nlp.stanford.edu/software/tagger.shtml)
[3] Stanford Named Entity recognizer http://nlp.stanford.edu/software/CRF-NER.shtml).
[4] MALTParser http://www.maltparser.org/userguide.html
[5] MPQA Subjectivity Lexicon http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/
[6] Arguing lexicon http://mpqa.cs.pitt.edu/lexicons/arg_lexicon/
[7] TiMBL https://languagemachines.github.io/timbl/
[8] Random Forest https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm