Detecting Stance in Tweets

Bhargav Mangipudi
Department of Computer Science
UIUC, Urbana, IL
mangipu2@illinois.edu

Vishaal Mohan
Department of Computer Science
UIUC, Urbana, IL
vmohan9@illinois.edu

Abstract

The aim of this project is to *identify stance* in tweets related to a topic given the target topic and the tweet text content. This is one of the tasks proposed for the **SemEval 2016** conference (Task 6a). For a given target topic, there are three possible stances for every tweet: `FAVOR`, `AGAINST` and `NONE`. The *140*-character restriction and the informal prose style makes *stance detection* more difficult and distinguished from regular sentiment analysis tasks. The main challenge is the inherent difference in vocabulary, lack of proper grammar structure and implicit display of stance. The best result that has been achieved for a similar task was 84 % accuracy reported by (Kiritchenko et al., 2014). We build a model with two stages: first to classify the relevance of the tweet to the target. The second stage classifier then predicts if the tweets that were classified by the first stage are `AGAINST` or `FAVOR` the topic. We achieve accuracies of 88.6 % and 74.4 % for the first and second stages respectively.

1 Introduction and Task Description

There is a sudden growth in the attention that social media and online forums are getting from the academic community and Twitter is probably the most popular among them. The reason could be because of relatively easier access to the data and the fact that the tweets say quite a lot about the reaction of people to changes around them. This lead to a spurt in the growth of research on sentiment analysis for tweets. The informal style of prose brings in new challenges to sentiment analysis. Tweets mostly tend to be limited in length, usually spanning one sentence or less. Also, they tend to have many misspellings, slang terms, and shortened forms of words.

The aim of this project is tangentially related to the problem of target-dependent sentiment analysis. Any event that takes place, immediately sees people taking sides on twitter and we aim to classify tweets as either AGAINST or FAVOR or not related to a given topic.

Formally, *stance detection* is the task of automatically inferring from text whether the author is *in favor* of the given target, *against* the given target, or whether neither inference is likely. Consider the target – tweet – stance example:

`Target`: Legalization of abortion
`Tweet`: A fetus has rights too! Make your voice heard.
`Stance`: FAVOR

This data instance is an example of a tweet being *in favor* of the given target topic.

We use the labeled data set provided by the SemEval 2016 Task 6a. The data set consists of 2,914 tweets whose stance is labeled. Also, these tweets are spread equally over the following *target topics*:

- Atheism
- Climate Change is a Real Concern
- Feminist Movement
- Hillary Clinton
- Legalization of Abortion

For the testing phase, we are given and tweet and its corresponding target and asked to identify the author's stance. Thus, this allows us to train different classifiers for each of the different participating targets.

It is important to note the difference between sentiment analysis and stance detection. A tweet that has a negative tone might actually be in favor of

the topic under concern. The task can be thought of as an aspect-based sentiment analysis problem for tweets where the aspect can be roughly derived from the given topic. The data is slightly skewed because the number of tweets which are neither FAVOR or AGAINST is close to 22 %. This skew might not seem like much for the overall task since we have three classes and the ideal split would have been 33%, but for the first task, we consider all the AGAINST and FOR examples to be of the same class and this skew will matter then. Many sentiment analysis models have features that depend on the number of retweets or the tweeter. Since the tweets in the given data-set are discrete and not tied to any user, we do not have access to more tweets in the conversation/context of the user.

The rest of the report is structured in the following manner: in Section 2 we talk about existing work related to this task. The next two sections describe the model used and experimentation details. Section 5 discusses the results we get and an analysis of the same and we end the report with a conclusion about our success, shortcomings and possible future direction.

2 Background / Related work

With the emergence of social media as the source of information for organizations and entities to estimate people's reaction to events, sentiment analysis has become a hot research topic among the Natural Language Processing community. There has been significant work on sentiment analysis of documents and more recently, of tweets and other short texts. A natural follow-up of sentiment analysis is target/aspect based sentiment analysis. These aspect dependent models have been extensively studied for Yelp's restaurant reviews, IMDb's movie reviews and Twitter data sets too. These algorithms have been found to perform pretty well in identifying aspects and then classifying the sentiment towards the aspects as positive or negative.

There are two main contrasting ways to perform target-independent sentiment analysis. (Turney, 2002) uses the mutual information between the phrase and two predefined seed words to classify tweets based on the semantic orientation of the sentimental phrases. In (Pang et al., 2002), the authors use machine learning algorithms to simply classify texts into three possible topics and this has proved to outperform unsupervised methods.

The task in hand is closer to a target-dependent sentiment analysis. The widely used method is to split the text using rules that identify possible aspects and then find the sentiment towards these aspects. There are many variants of classifying using the rule-based approach (Jiang et al., 2011).

Target dependent classification is considerably harder for tweets when compared to reviews. In reviews, an aspect and a sentiment expressed in the same sentence means that we could say with a high confidence that they are related. This is not true for tweets. Moreover, the short nature, grammatical incoherence, emoticons and abbreviations make the problem all the more harder.

There has been very limited work attempting to do stance detection on tweets. The most recent and related work related to target dependent classification was in (Jiang et al., 2011) where the authors use rule-based methods and graph optimization to classify sentiments towards a target. In (Somasundaran and Wiebe, 2010), the authors attempt stance detection on online debates which is a lot more formal than twitter.

3 Model

This section describes in brief the model we use to classify stance. We design a two-step approach where:

- In the first stage, we classify the relevance of the tweet to the target: NONE vs RELEVANT = FAVOR + AGAINST.
- In the second stage, we use the example that are classified as not None and classify them as either AGAINST or FAVOR the topic.

For the first step, we experimented with different features like One hot representation of words, Bag of Words (using word vectors) and the details about the exact features and the reason for choosing them have been explained in the next section along with the results obtained for every attempt. For the second step, we make use of existing sentiment lexicons, NRC-Canada (Mohammad and Turney, 2013) and Sentiment140 (Go et al.,) to add features like

- Sum of all the positive sentiment scores from the lexicon, sum of the negative scores.

- The vector representation of the word that gives the maximum positive and negative sentiment score.
- We also create a parse tree to find the noun that is associated with the word with the extreme positive and negative words and add the vector representations of these words as well.
- The final feature is the sum of the vectors of the word in the target.

In each step, we perform binary classification using a linear classifier. We talk more about other details of the model in the Experiments section.

4 Experiments

This section contains the meat of the report that talks about the exact experiments we carried out to choose the best parameters for the model.

4.1 Data set Analysis

Target	FAVOR	AGAINST	NONE	Count
Atheism	92	304	117	513
Climate...	212	15	168	395
Feminist...	210	328	126	664
Legalization...	121	355	177	653
Hillary...	178	393	178	689
Total				2,914

Table 1: Number of tweets per topic – categorized by Stance

The use of word-embedding for higher-dimension representation of text content is prevalent in the recent years due to their effectiveness in compositional semantics. This idea was popularized by a seminal paper by (Bengio et al., 2003), which provides deep insights into why word embedding are powerful. The `word2vec` project by (Mikolov et al., 2013) is used highly for practical purposes. For our project, we use the GloVe word embedding provided by (Pennington et al., 2014) at Stanford Core NLP team. These embeddings are trained on 27 billion tokens collected from 2 billion tweets.

Dataset Vocabulary Size	8,827
Words Not Present in GloVe embeddings	2,660
Absent Hastags count	1,875
Absent common words count	785

Hashtag	Total Occurrences
'#GamerGate'	56
'#LoveWins'	44
'#God'	42
'#WakeUpAmerica'	37
'#HillaryClinton'	31
'#SCOTUS'	31
'#ProLifeYouth'	24
'#YesAllWomen'	22
'#feminist'	21
Unique Hashtag Count	1,875

Table 2: Hashtags not present in GloVe Word Embeddings

On preliminary analysis we found that a majority of hashtags from our data set did not have a representation in GloVe trained on Twitter tokens (Table 2). We found 1,875 unique hashtags across the corpus that were not present in the GloVe embeddings.

In table 3, we look at the common words that were not present in GloVe, most of them are *contractions* of English words (like `don't`, `can't`) and numerals. Cardinal values that appear in the tweet do not add semantic meaning to our work. For most contractions, they were absent from the GloVe embedding but were present in the sentiment analysis lexicon that we considered. Thus, they are included in the Part 2 task.

Common words	Total Occurrences
'don't'	166
'...'	156
'it's'	137
'i'm'	120
'can't'	76
'2'	52
'you're'	46
'doesn't'	44
'..'	41
'women's'	29
'that's'	28
'she's'	28
Unique Words Count	785

Table 3: Common words not present in GloVe Word Embeddings

4.2 Environment

All the experimentation work was performed in Python (2.7) using the NLTK (3.1) toolkit library by (Bird et al., 2009) for tokenization and related tasks. We use the popular ML library scikit-learn (0.17) by (Pedregosa et al., 2011) for all classification tasks. We used Jupyter Notebook (erstwhile IPython) to help make the analysis simpler and efficient.

4.3 Preprocessing

Due to the informal nature of tweets, the level of preprocessing can have a major impact on the performance of the classifier. Also, if we try to use word vectors for our features, a spelling error, two words combined into one will not count towards the feature as it won't be present in our dictionary for the word vectors.

- Since hashtags are a major indicator for target topic *and* stance, we had to treat hashtags in a better way. By looking at some examples, we says that some hashtags are amalgamations of words, for example: `#LoveWins, #WakeUpAmerica` – which convey important information. Thus, we use two heuristics to split hashtags into individual components. First, we try to split hashtags using the *capitalized* form of some hashtags. Second, for hashtags that cannot be processed using the prior process, we use a Dynamic Programming approach to get the best split into meaningful words (that fit into the English dictionary).

  "big #awesome-dayofmylife because #iamgreat" `is transformed to` "big awe some day of my life because i am great"

  Even though there are some caveats to above approach, like #GamerGate is a single entity representing the recent issues of harassment against women in the gaming industry – #GamerGate is related in sense to 'Gamer' but not 'Gate', we see a general improvement in classification accuracies after adding hashtag splits to the list of features. Out of the *3846* total hashtags present, we were able to split *1975* hashtags successfully into meaningful components – thus enriching our tweet content.

- We also convert all the words into lower case after splitting the hashtags for consistency as the GloVe words has words in lower case.

- We also add the original hashtag before split to the tweet. This is because the sentiment lexicons that are obtained from twitter corpora has hashtagged words.

- We also use the CMU NLP Tweet tool (Noah's ARK) (Owoputi et al., 2013) which has a tokenizer to annotate tweets for POS tags. This particular POS-tagger is again trained on Twitter corpora and has more extensive tags like 'Abbreviations', 'Hashtags' and 'Exclamations'. These tags are used in the first step to remove words that do not matter when classifying relevance to the topic: determiner, conjunctions, punctuation. In the second step, we use the POS tags to identify the noun that a word with a particular sentiment score describes.

4.4 Part 1 - Target Relevance Classification

The first part of our project relates to finding relevance of a tweet to a given target. For this part, we use the constituent tokens of a tweet to try to classify how likely the tweet is to relate to the target. Thus, we combine the labeled stance into two categories : { `NONE, FAVOR + AGAINST`} and model the problem as a binary classification over the new labels. Since there dataset is skewed with each target having about 20 % tweets, we augment the dataset for each target and increase the amount of `NONE` tweets for a target by taking the union of `NONE` tweets for all given targets as they are all irrelevant to any of the given targets. This trick gives use a more *balanced* data set for each target. This has been done under the assumption that the tweets marked as 'None' for one topic is not related to any of the other topics either and this is a safe assumption looking at the dataset.

After the steps mentioned in pre-processing, we use the remaining token constituents, extracted for each of the tweet, for our experiment. We test two different approaches to this task: a bag-of-words over the vocabulary using One-Hot encoding and the second one using 200 dimension GloVe word-embeddings trained over 27 billion twitter tokens. We use the SVM classifier with the regularization parameter = 1, and a 'linear' kernel. We do hyper-parameter tuning for selecting the regularization

parameter and a stratified 5-fold cross-validation for evaluating the performance measure. Table 4 and 5 show the results obtained for each of the 5 targets using the two approaches described above. For the One-Hot encoding, we use the minimum document frequency parameter of **3** and thus, we get an average vocabulary size of 942 words. Each tweet is presented by an **0 - 1** vector with **1** for all words that are present in the tweet. It is important to note that the One-Hot feature vector changes with target because we create the vocabulary by setting a threshold on the number of occurrences of the particular word. The size of the features is different for each target as the vocabulary of words observed is different for each target.

Target	# Features	Accuracy	F-1
Atheism	900	**0.909**	**0.930**
Climate...	950	0.895	0.932
Feminist...	1000	0.900	0.915
Legalization...	840	0.870	0.892
Hillary...	1015	*0.852*	*0.877*

Table 4: Averaged Scores of Stratified 5-Fold CV using a One-Hot bag-of-words representation

Target	# Features	Accuracy	F-1
Atheism	200	**0.908**	**0.930**
Climate...	200	0.881	0.923
Feminist...	200	0.880	0.898
Legalization...	200	0.873	0.896
Hillary...	200	*0.836*	*0.860*

Table 5: Averaged Scores of Stratified 5-Fold CV using Sum of GloVe vector embedding

For the GloVe embedding approach, the size of the feature vector is constant across targets as it is the sum of the word vectors of all the words in the tweet. Since, we remove stop words from the tweets in pre-processing, we are left with words that are related to the gist of the tweet – which should in-turn reflect the target topic.

4.5 Part 2 - Tweet Stance Classification

We use features as described in the Model section. For simplicity, we calculate the accuracy for the second stage separately, assuming that the first stage does a perfect classification. We then use the statistics mentioned so far to calculate a theoretical accuracy for the entire model.

Target	Accuracy Score	F-1 Score
Atheism	0.777	0.526
Climate...	**0.916**	**0.955**
Feminist...	*0.620*	*0.529*
Legalization...	0.720	0.494
Hillary...	0.686	0.416

Table 6: Averaged Scores of 5-Fold CV for `FAVOR vs AGAINST` classification

5 Results

For the first stage, we obtain an average accuracy of **87.5 %** when we use the sum of the word vectors as a feature and **88.6 %** when we use a one-hot feature representation. The higher accuracy for the One-Hot representation might be attributed to the fact that occurrence or non-occurrence of certain words can actually add knowledge about whether we are talking about the topic or not. We obtain excellent F-1 scores as well for the first stage, again Atheism being the best performing topic, mostly owing to the occurrence of similar words across all tweets. For example, the occurrence of the word 'rain' will immediately mean that it is related to the topic of 'Climate'. The worst performing target is 'Hillary Clinton', which can be explained by the very specific nature of the topic. If the tweets talk about proper nouns, none of are features are engineered to capture this as we do not use any external information.

For the second stage, we obtain a considerably lower accuracy of **74.4 %**. The best performing topic is the one on 'Climate' and the worst performing topic is the one about 'Feminism'. The poorer performance of the stance detection could be because of lack of better features. We assume that the noun that occurs closest to the words with extreme sentiments are the aspects that we talk about. Note that this is assuming we classify perfectly on the first stage.

We calculate the theoretical accuracy for the entire model in the following way: Let us assume that we have 100 examples. The first step's accuracy is 88.6 % which means that we make mistakes on 11.4 examples. Of the 88.6 examples that remain, we assume that 22% of the examples are 'None' going by the skew in the data set. So the examples we classify correctly is `(0.22 * 88.6) + (0.744 * 0.78 * 88.6)` = 70.9. Therefore, a conservative estimate of the overall accuracy of the model comes out to be **70.9 %** .

6 Conclusion

When we compare our results to the only closely related work on Target Dependent Sentiment Analysis, we notice that we beat their overall accuracy by 2%. However, this might not be an entirely fair comparison owing to the small size of the dataset we had access to. We also claim that we can make improvements to the stated accuracy if we add more meaningful features that gives an estimation of how close the words in the tweet are with the target.

This particular task of detecting stance is not trivial as we see a lot of different forms of expression of stance that are implicit and/or implied and not gleanable from the surface tweet. For example: one common way to express *negative slant* towards a topic is to do a comparison with other competitive topics without explicit referral – e.g. "Jeb Bush is the best candidate for presidency." shows an implied *AGAINST* stance towards *Hillary Clinton*. Here, stance detection is mostly driven by knowledge-base representation and reasoning inferences along with sentiment analysis. Our approach to the problem led to some favorable results but we still believe that by adding more knowledge-system based features, there is some scope for improvement.

The other major model that is gaining traction these days is neural networks for sequence processing and semantic analysis. Project peer feedback referred us to the work done by (Socher et al., 2013) on RNTN (Recurrent Neural Tensor Network) based approaches which tries to use networks based on dependency tree to evaluate sentiment scores for sentences. We would like to try this approach for future work and compare results with our analysis.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March.

S. Bird, E. Klein, and E. Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

Alec Go, Richa Bhayani, and Lei Huang. Twitter Sentiment Classification using Distant Supervision. Technical report, Stanford University.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 151–160, Stroudsburg, PA, USA. Association for Computational Linguistics.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M. Mohammad. 2014. Sentiment Analysis of Short Informal Texts. *J. Artif. Intell. Res. (JAIR)*, 50:723–762.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *Nips*, pages 1–9.

Saif M. Mohammad and Peter D. Turney. 2013. Crowdsourcing a word-emotion association lexicon. 29(3):436–465.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*, EMNLP '02, pages 79–86, Stroudsburg, PA, USA. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Swapna Somasundaran and Janyce Wiebe. 2010. Recognizing stances in ideological on-line debates. In *Proceedings of the NAACL HLT 2010 Workshop on Computational Approaches to Analysis and Generation of Emotion in Text*, CAAGET '10, pages 116–124, Stroudsburg, PA, USA. Association for Computational Linguistics.

Peter D. Turney. 2002. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 417–424, Stroudsburg, PA, USA. Association for Computational Linguistics.