Stance Detection in Tweets

Rahul Huilgol, *rrh2226*
University of Texas at Austin
rahulrh@cs.utexas.edu

Ashwini Venkatesh, *av28895*
University of Texas at Austin
ashuven6@cs.utexas.edu

Abstract

In this paper we attempt to solve the task of stance detection on tweets using several features extracted from the tweet. We study the importance of various features, and the performance of different kinds of machine learning architectures.

1. Introduction

Stance detection can be defined as the task of automatically determining from text and a given topic, whether the author of the text is in favor of the topic, against the topic or neither. For example, consider the tweet: " A foetus has rights too! Make your voice heard". We humans can easily deduce that this tweet is against the target of "Legalization of abortion".

In this work, we aim to build a system that can determine this stance from tweets. Tweets present an additional challenge over general text, in that they are short. To successfully predict the stance, the system would have to identify information that may not be present in the text itself. For example in this case, it has to connect that supporting the rights of foetus is on the same side as being against abortion.

Stance detection has lot of applications and is closely related to tasks like information retrieval, text summarization, textual entailment. It can be seen as an extension of sentiment analysis or opinion mining, where instead of general sentiment, the goal is find the author's opinion on a particular topic. This task is part of the International Workshop on Semantic Evaluation SemEval-2016. They provide a dataset in the form of tweets labelled with the topic and stance.

2. Related Work

The problem of stance detection has not seen much work in literature. Some work that exists focuses on using large bodies of text like essays and debates, but nothing for short text like tweets. Works like [1], [12], [13], [3] propose solutions for stance detection in the context of online debates. [1] evaluates rule based and Naive Bayes classifiers over features like unigrams and bigrams, repeated punctuation, syntactic and generalized dependencies, and features extracted from the tool Linguistic Inquiry and Word Count (LIWC) - words per sentence (WPS), pronominal forms (Pro), and words with positive and negative emotions (PosE) and (NegE). Evaluation shows that these features do not give any significant improvement over baseline. The LIWC feature set is the only feature set that appears to show some improvement over a some set of topics. In [12] associations that are indicative of opinion stances in debates were learnt by mining the web. This knowledge is combined with discourse information and the debate side classification task is formulated as an Integer Linear Programming problem. One significant difference of this existing work from our work is that we have to deal with the additional challenge that tweets have less textual information compared to essays of debates. [2] discusses an approach for stance classification on student essays using two sets of features, part of speech generalized dependency subtrees and capturing the relationship between words in prompt (topic) and essay using Wikipedia cross-references to compute a similarity metric. But his ideas likely dont translate well to tweets, which typically dont have great grammatical structure and tweets also have less number of words talking about the topic, making it very easy to miss similarities of topic and tweet.

Sentiment analysis is a related problem which is well studied. [9] builds a set of SVM classifiers. This approach builds 2 types of word-sentiment lexicons, sentiment word hashtags and emoticon lexicons. Apart from this they use the general features like ngram to train the model. The huge size of the lexicon helps in training a good mapping from a tweet to its sentiment. [6] is similar to [9]. Additionally, to adequately capture the sentiment of words in negated contexts, a separate sentiment lexicon is generated for negated words. In stance detection, the challenge lies in the fact that the target of the tweet may not even be present in the tweet. We cannot directly correlate the sentiment expressed in the tweet to the stance making this a harder problem than sentiment detection.

3. Problem Statement

We formally define the problem statement as the following. Given a tweet and a topic, the task is to predict whether the tweet is taking a stance on the topic, and if so whether

this stance is in favor of or against it. Thus the three labels which form possible predictions are FAVOR, AGAINST, NONE. Here NONE represents that the tweet does not take any stance on the topic. It is worth pointing out that a positive or a negative sentiment does not correspond to a favor or against stance.

4. Dataset

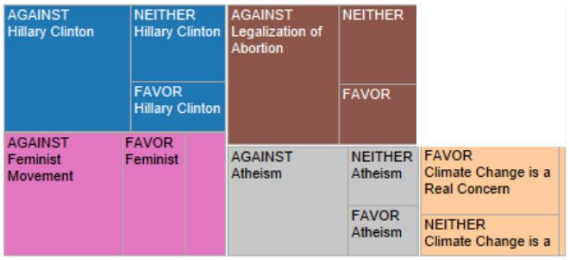

Figure 1: Visualization of the stance of tweets in dataset for each topic

We used the dataset provided by the problem's SemEval-2016 task. Training data consists of 2915 labeled samples for five topics: *Atheism*, *Climate Change is a real concern*, *Hilary Clinton*, *Legalization of Abortion* and *Feminist Movement*. Each sample comes with the text of the tweet, a topic and the stance taken by the tweet on that topic. Two other features are provided with each sample: its sentiment, and whether it contains opinion about the topic, opinion about something else, or no opinion. These features are for understanding how the model is performing not to use for training. 1956 samples with gold standard labels are also provided for testing. Figure.1 shows the distribution of data across different labels of stance for each target. We can see that the data set is pretty unbalanced and has almost 50% of the tweets labeled as AGAINST. The number of samples for NONE is less as well.

5. Technical Approach

The idea we pursued was to model this as a multi-class classification problem. We built different kinds of machine learning classifiers that uses variety of features as described below.

5.1. Feature-sets

We first describe the features used to train the model for the task. Experiments used different combinations of these features. Each of these features actually give a feature vector, i.e. they form a feature set.

- **Words**: (Indicated in results as *'Words'*) Bag of words representation of the tweet, in the form of a one-hot vector of length equal to the number of unique words in the corpus (both training and test data).

- **Continuous word representations**: Since the number of words in tweets is less, and the dataset is small, we sought to augment the feature set with features continuous word representations learnt from a larger dataset. We experimented with two types of such representations. For both these types of embeddings, a tweet was represented as the average of vectors of all words in the tweet.

  - **Word2vec embeddings**: (*'Words2vec'*) A word is represented by its neural word embedding using Word2vec[7] models trained on the Google News dataset. This feature has been indicated in results as *'Words2vec'*.
  - **GloVe**: (*'GloVe'*) GloVe which stands for global vectors for word representation, is based on factorizing cooccurence matrix of data[11]. We used models that were trained on a large corpus of tweets to give a vector of 100 dimensions.

- **Topic**: The target topic of the tweet. We experimented by using 3 representations for this feature. The first is a one hot vector of length equal to the number of topics (*'Topic1hot'*), second is just one categorical label for the topic (*'Topic'*), and third is as the vector using word2vec embeddings of topic (*'Topic2vec'*).

- **Lexicons**: To augment the limited information in tweets, we used lexicons. MPQA Subjectivity lexicon provided Polarity and Subjectivity for around 8000 words[14], and Opinion lexicon[4] which provides sentiments of around 6000 words. These lexicons thus provide three sets of features represented in two ways.

  - Naive approach:(*'LexiconsNaive'*) For each word in the tweet, its subjectivitypolaritysentiment was given if present in lexicon. If a corpus word is not present in tweet or not present in lexicon, a default value was given. This means each of the lexicon gives us features of length equal to size of corpus.
  - Clustered vectors:(*'LexiconsClustersDistance'*) However, since the size of the lexicon is large, above approach greatly increases the dimensionality of the feature vector given as input. We then tried to build a smaller feature using the following steps. Find the word2vec embeddings of all words in a lexicon and cluster each label (for example: positive, negative, neutral for sentiment) of the lexicon into

20-30 clusters based on the vector embeddings. This essentially groups words in lexicon by similarity. An example of a cluster formed this is: ['entranced', 'wowed', 'enchanted', 'dazzled', 'wowing', 'enthralled', 'enraptured', 'awestruck', 'smitten', 'rapt', 'spellbound', 'mesmerized', 'awed']. Now compute the distance of each word in the tweet to the centroid of the clusters and build a feature vector. Average of all these distance vectors is the feature set generate for each tweet. This feature set for each lexicon has length equal to number of clusters. Thus we are reducing the length of the feature vector used and also making them more informative.

- **Part of Speech tags**:(*'POStags'*) We used a Twitter specific POS tagger[10] to assign each word one of 25 tags. The number of words assigned to each of these 25 tags in a tweet is used to represent this feature-set.

- **Top N-grams** This feature-set was used as an alternative to having one feature for each word in corpus as part of the 'Words' feature. Here, we compute the top 500 unigrams (*'Top1grams'*) and bigrams *'Top2grams'* in the dataset and create a feature vector with the frequency counts of these top ngrams for each tweet. This feature was tested with both inclusion and exclusion of stop words.

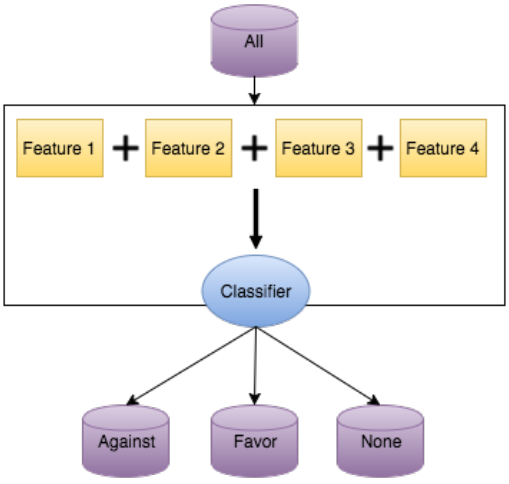

Figure 2: Simple model trained on concatenated feature sets

5.2. Model Architecture

Following are the different models with which were used to predict the stance of a tweet using sets of features above.

5.2.1 Simple model

This is the simplest model, in which we just concatenate some feature sets chosen to form a single feature vector for each sample. One classifier is built to learn from these training samples.

5.2.2 2 Layer Model

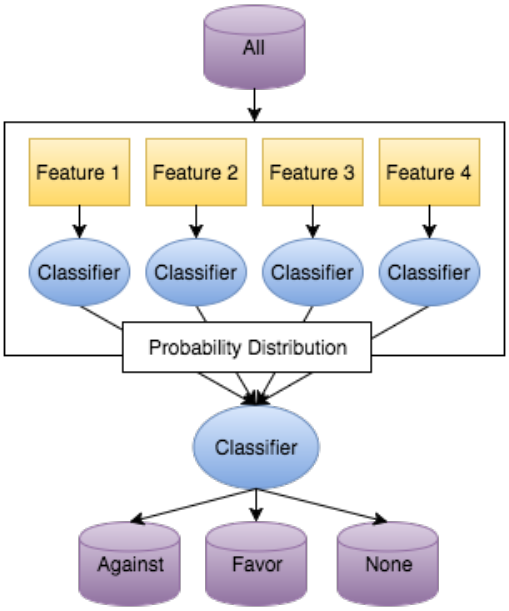

Figure 3: 2 Layer stacked model in which second classifier uses decisions made by first classifier as input

In this model we train a classifier for each feature set to predict over all labels. We take the predicted probability distribution over all labels from each classifier and combine this to form a feature vector as shown in Fig. 3. This feature vector is then used to train next level of classifier to predict over all labels. This model thus learns the weights which should be given to each feature used to train in the first level. This method should typically give better performance than considering all features to be equal.

5.2.3 2 Stage Model

In this model we train two classifiers and divide the task of prediction into two stages as indicated by Fig. 4. The first model is trained to differentiate tweets with stance in them from tweets which dont. This classifier predicts whether the label is STANCE or NONE, that is the variables FAVOR and AGAINST are merged into a single variable STANCE for this classifier. The second classifier is trained only on those tweets which have some stance in them, to predict FAVOR or AGAINST. The intuition behind this is to make

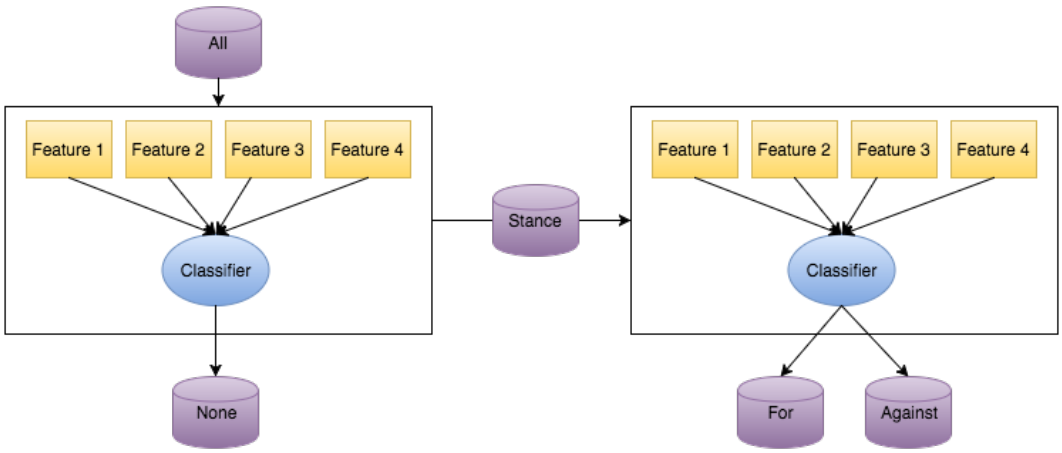

Figure 4: 2 Stage model which has a separate classifiers for each stage

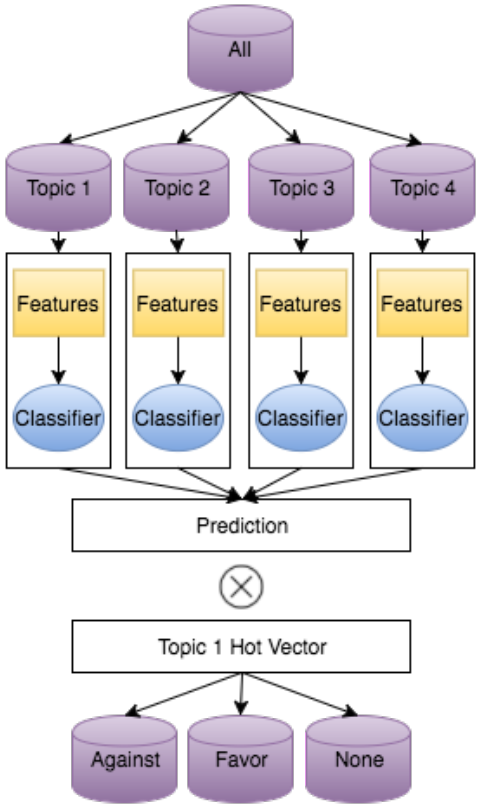

Figure 5: Topic wise model

the task for second classifier easier by having it not ignore tweets which do not have stance in them. At test time, the predicted NONE labels are considered from the first classifier's output. The prediction for samples which first classifier predicts as STANCE are t In the first classifier we train the model to differentiate between tweets which take a stance and one which doesn't. We pass on the tweets which take a stance to next level where you classify whether the tweet is for or against the target.

5.2.4 Topic wise Model

We built one model in which we used a different classifier for each topic. This classifier used all features, and predicted Favor, Against or None. At test time, we gave all test samples to each topic trained classifier. But using the topic 1-hot vector, we merged these predictions in such a way that for the final prediction of a sample, we used the classifier for the topic which a particular sample belonged to. The idea behind this was that each of the topics are very different, and features which are useful for one topic may be significant for the other topics. Separating the classifiers should make it easier to train.

5.3. Classifiers

We experimented with the following classifiers:

- Support vector machines (SVM) with Linear kernel: SVM is a very popular classifier used for text classification. [5] details various advantages that SVM hold for text classification. They are robust classifiers and work well with the high dimensional sparse input feature vectors that text has. Because of the high dimensional input space, a linear kernel works best. This is because in this high dimensional space, there exist huge options for the separating hyperplane to be learnt. There is no need to use any other kernel to map to higher dimensions.
- Logistic Regression has been used in the 2 Layer model for the classifier in the second layer. The intuition behind this was that it essentially tries to learn appropriate weights for different features given. In this case, we wish to learn weights with which to combine

the predictions of classifiers trained on different feature sets.

- XGBoost which is a form of gradient boosting algorithm, trains an ensemble of trees and has been popular for text classification of late.

6. Preprocessing

The preprocessing performed on the tweets text was performed in the following manner:

- Convert text to ascii by removing non-ascii characters
- Expand the hashtags in tweets by splitting them into their constituent words. For example, #HilaryClinton splits into 2 tokens as #Hilary #Clinton
- Remove features which do not have significance for stance classification: convert all numbers in the text to 'num', convert tweet to lower case, convert twitter usernames of the form @username to a standard AT_USER, and remove stopwords
- Tokenize the tweets using Tweet Tokenizer provided by the nltk package

7. Experiments

The experiments were done in Python, using the library Scikit-learn for machine learning models, and nltk library for natural language processing tasks. We performed many experiments using different combinations of the set of features above, and the different model architectures above. Grid Search used to optimize the hyper parameters for the classifiers.

7.1. Evaluation Criteria

The accuracy of predicted stance labels is a decent metric but is not as informative as precision, recall and F-1 scores. These metrics were computed for both FAVOR and AGAINST labels, as well as the Macro F-1 score. Please note here that the Semeval task is not aimed at measuring accuracy on NONE. It is just a category that the model can assign if some tweets have neither stance in them, so that precision and recall of stances is not affected.

7.2. Baselines

We built baseline models which basically had a single classifier that is only given as input features bag of words of the tweets. The classifiers Multinomial Naive Bayes and SVM were used for baseline. We see from Table. 1 that the best Macro F-1 score for the baseline is **0.5817**.

7.3. Results

7.3.1 Simple model

In this model which concatenates feature sets to train a single classifier, we first tested how different features were performing individually. All observations use a SVM classifier. XGBoost also provided similar performance, +- 1%.

- We see from Table. 2 that none of the other features are as discriminative as the topic of the tweet.
- Another interesting observation was that representing Topic as a 1-hot vector (5 dimensions) gave an increase of 0.05 over representing it as a categorial label (1 dimension).
- Top1grams and top2grams give F-1 score lot lower than other features. The effect of POStags towards F-1 score was pretty negligible.
- Word2vec performs significantly better when used alone than GloVe. But using them together leads to better than either of them used alone.
- Topic2vec (representation of the topic using word2vec) performs better than topic1hot when used with Word2vec, because it allows the classifier to compute the similarity of words in tweet and words in topic.
- Including the LexiconsClustersDistance feature actually seems to confuse the model and give worse results than without. One reason could be that the model is finding it hard to learn how to map the sentiment, polarity and subjectivity information with the words in tweet. This is because the presence of positive sentiment for example is only useful when the model is able to detect that the tweet has an opinion about the topic. The LexiconsNaive feature set provides the lexicon score of all tweet words that are in corpus. But this blows up the dimensionality of the problem, potentially making the task harder for the model.
- We also start to see a general trend in the results, since the training data had more samples with the stance AGAINST, the models are better at classifying tweets with AGAINST stance. Both precision and recall (and hence F-1 score) of AGAINST are significantly higher than those for FAVOR.
- The best F-1 score we got with this model corresponded to when the topic feature was included either as a vector using Word2vec or as a 1 hot vector, **0.6522**. Including other features along with topic does not seem to improve the score. The best accuracy we got was when we used the feature set: Words as GloVe, Words2vec, Topic2vec and POStags, **0.6637**.

Model	Accuracy	Favor			Against			Macro
		Precision	Recall	F-score	Precision	Recall	F-score	F-score
Naive Bayes	0.5925	0.4502	0.4605	0.4553	0.6925	0.7245	0.7081	0.5817
SVM	0.5596	0.4244	0.4803	0.4506	0.7395	0.6154	0.6718	0.5612

Table 1: Baseline results with Naive Bayes and SVM using Bag of words features

Feature sets	Accuracy	F-1 scores		
		Favor	Against	Macro
GloVe	0.6141	0.3816	0.7511	0.5663
Words2Vec	0.6293	0.4591	0.7511	0.6051
Topic2Vec	0.6621	0.5201	0.7844	0.6522
Topic1hot	0.6621	0.5201	0.7844	0.6522
LexiconsClustersDistance	0.6189	0.4757	0.7429	0.6093
Top1grams, Top2grams	0.5917	0.3043	0.734	0.5192
Words, LexiconsNaive,Topic, POStags	0.6037	0.4398	0.728	0.5839
Words, LexiconsNaive,Topic1hot, POStags	0.6541	0.4881	0.7812	0.6347
Words2vec, Topic1hot, POStags	0.6621	0.5201	0.7844	0.6522
Words2vec, LexiconsClustersDistance, Topic1hot, POStags	0.6549	0.4706	0.7812	0.6259
Words2vec, LexiconsClustersDistance, Topic2Vec, POStags	0.6597	0.4989	0.7845	0.6417
GloVe, Topic1hot, POStags	0.6557	0.4516	0.7844	0.618
GloVe, Words2Vec, Topic1hot, POStags	0.6573	0.4587	0.7834	0.6211
GloVe, Words2Vec, Topic2Vec, POStags	0.6637	0.4989	0.7864	0.6427

Table 2: Accuracies and F1-scores with different featuresets for the simple model which uses featureset concatenation

7.3.2 2 Layer model

Table 3 shows selection of results we obtained using the 2 layer model using SVMs. Results are about the same as using a single model. In some cases, there is a slight increase in F-1 score, for example when using the features sets Words2vec, POStags, LexiconsClusterDistances, Topic1hot gives a slightly higher F-1 score but slightly lower accuracy. The worse performance of 2 layer model compared to a single SVM can perhaps be attributed to the fact that the detection of stance needs valuable information that is a combination of the feature sets we devised.

7.3.3 2 Stage model

Table 5 shows the results of using a 2 stage model with SVMs. We see that the first stage classifier is only able to recall about half of the samples, and has very low precision. We can also see that the second stage classifiers has significantly higher precision and recall, and hence higher F-1 scores. The best 2 stage model as shown had F-1 scores of 0.87 for AGAINST and 0.60 for FAVOR. The best F-1 scores of models earlier was 0.78 and 0.52. This means the second stage classifier is able to classify FAVOR and AGAINST much better than a single classifier. But the inability of the first stage classifier to detect NONEs well holds back the final performance, such that the final F-1 score is lower than before. The problem faced by the first stage classifier could be that the distribution of labels STANCE and NONE in the training dataset is very skewed. There are only a few samples with the label NONE. We tried training the first classifier with equal number of STANCE and NONE samples by randomly sampling samples with STANCE variable. But this still did not yield good results because this does not solve the issue of small training data for the class label NONE.

Feature sets	Accuracy	Favor			Against			Macro
		Precision	Recall	F-score	Precision	Recall	F-score	F-score
Words2vec, POStags, LCD	0.6141	0.5336	0.4441	0.4847	0.7147	0.7566	0.7351	0.6099
GloVe , POStags, Topic1hot	0.6559	0.6914	0.3684	0.4807	0.6957	0.8951	0.7829	0.6318
Words2vec, Topic2vec, POStags, LCD, Top1grams, Top2grams	0.6357	0.5127	0.5296	0.521	0.7206	0.772	0.7454	0.6332
Words2vec, POStags, LCD, Topic2Vec	0.6501	0.5774	0.4539	0.5083	0.7314	0.8112	0.7692	0.6388
Words2vec, POStags, LCD, Topc1hot	0.6501	0.5774	0.4539	0.5083	0.7323	0.8112	0.7697	0.639

Table 3: Results for the 2 layer model (LCD stands for LexiconsClustersDistance

Accuracy	Favor			Against			Macro
	Precision	Recall	F-1 score	Precision	Recall	F-1 score	F-1 score
0.6533226581	0.5661	0.5493	0.5576	0.7402	0.7692	0.7545	0.656

Table 4: Best results with topic wise model for the feature set: Words2vec, LexiconsClustersDistance, Topic1hot

Classifier	Label	Precision	Recall	F1-score
Stance vs None	NONE	0.35	0.49	0.41
	STANCE	0.87	0.79	0.83
	avg	0.78	0.74	0.75
Against vs Favor	AGAINST	0.81	0.93	0.87
	FAVOR	0.75	0.50	0.60
	avg	0.79	0.80	0.79
Final	AGAINST	0.72	0.81	0.76
	FAVOR	0.59	0.24	0.34
	NONE	0.35	0.49	0.41
	avg	0.62	0.61	0.59
	Accuracy	**0.6629**		

Table 5: Two Stage model with the feature sets: Words2vec, Topic2vec, POStags, LexiconsClustersDistance, Top1grams & Top2grams

7.3.4 Topic wise model

In this model we train one classifier for each topic. Results are shown in Table .4. This gives a slight increase in the Macro F-1 score over the previous best. This makes **0.656** as the best F-1 score we achieved in the project. The F-1 score for FAVOR has increased by 3%. The fact that a topic wise model works slightly better than a simple model is evidence that the topics do not share many characteristics between them for the task of stance classification. Some topics might even have characteristics which are contradicting, which would explain the slight increase shown by topic wise model.

7.3.5 Effect of increase in training data size

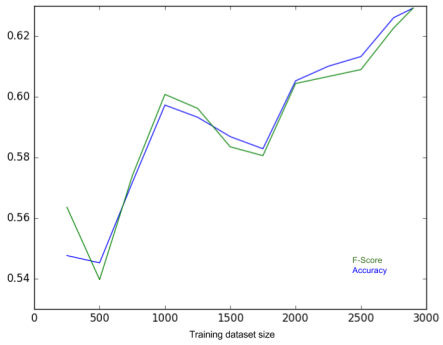

Figure 6: Effect of training dataset size on accuracy and f-score

We can see in Fig. 6 that the accuracy and F-1 scores of the model sharply increases as we increase the size of the dataset. Even towards the end of the graph, the curve maintains a steep slope. This indicates that the model is nowhere saturation in terms of how much it can learn, it is probably underfitting. This means that it can learn and improve significantly if its given more data.

7.3.6 Using additional features

As mentioned earlier, the SemEval task provides additional features to analyze the problem and models. We tested a simple SVM model with the features topic, sentiment, and whether the tweet contains opinion about target, opinion about something other than targer, or no opinion. Using these features it would seem that for a general case, a model should be able to figure out the stance of the tweet. But on this dataset, we could only obtain an accuracy of 0.7278 and a Macro F-1 score of 0.6914. This is not very high when compared to the scores we achieved. This is an indication that this a pretty hard problem.

8. SemEval task participants results

The task's evaluation period was in January, so we couldnt make a submission. The paper deadline however was around early April. Results were released recently [8]. 19 teams participated, and the highest classification F-1 score obtained was 67.82. This is only a couple of percentage points higher than the highest we obtained. The fact that the highest in this task was still pretty close to the baseline models, shows that this is hard task and current best practices are nowhere close to achieving good performance on this task.

9. Future work

Couple of things that made this task hard for us, was that the labeled data available was not much to train a supervised machine learning classifier well. One direction which can be pursued is to collect more unlabeled data from twitter with hashtags related to the topic, and use semi supervised learning approaches. Subjectivity and sentiment of the tweet provide important clues to stance detection. The lexicons we used did not prove to be very useful. We need to develop lexicons that better capture stance and sentiment of informal short text. We also need some component of additional knowledge base so that we can find the similarity of words in tweet to the topic. This was the idea behind our usage of distributional representations like Word2vec and GloVe, but we need more specialized approach. Building a dataset of hashtags which are labelled with their stance would also be very useful.

10. Conclusion

The above experiments show that the feature that is most helpful for this task is the topic of the tweet. We also see that we cannot generalize stance detection for every topic. Every topic has different vocabulary and different form of opinions which are used to express for or against the target. This is why training a different classifier for each topic is a good idea. However if the topics are similar like 'feminism' and 'legalization of abortion' we can transfer features learnt from one model to other. Current performance on thsi task shows that there is a huge scope for improvement, and a need to devise more novel features to capture the stance of a tweet.

References

[1] P. Anand, M. Walker, R. Abbott, J. E. F. Tree, R. Bowmani, and M. Minor. Cats rule and dogs drool!: Classifying stance in online debate. In *Proceedings of the 2Nd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, WASSA '11, pages 1–9, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[2] A. Faulkner. Automated classification of stance in student essays: An approach using stance target information and the wikipedia link-based measure. In *The Twenty-Seventh International Flairs Conference*, 2014.

[3] K. S. Hasan and V. Ng. Stance classification of ideological debates: Data, models, features, and constraints.

[4] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[5] T. Joachims. *Text categorization with support vector machines: Learning with many relevant features*. Springer, 1998.

[6] S. Kiritchenko, X. Zhu, and S. M. Mohammad. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, pages 723–762, 2014.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[8] S. M. Mohammad, S. Kiritchenko, P. Sobhani, X. Zhu, and C. Cherry. Semeval-2016 task 6: Detecting stance in tweets.

[9] S. M. Mohammad, S. Kiritchenko, and X. Zhu. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242*, 2013.

[10] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, and N. A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics, 2013.

[11] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation.

[12] S. Somasundaran and J. Wiebe. Recognizing stances in online debates. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 226–234. Association for Computational Linguistics, 2009.

[13] D. Sridhar, L. Getoor, and M. Walker. Collective stance classification of posts in online debate forums. *ACL 2014*, page 109, 2014.

[14] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 347–354. Association for Computational Linguistics, 2005.