Stance detection for Tweets using various Computational Linguistic Approaches

Apurva Gupta, Surbhi Paithankar, Sandra Keubler

***Abstract*—Any given sentence may have an attitude attached to it. It is very easy for humans to interpret the stance given a sentence. However, a computer needs to be trained for understanding and detecting the stance towards a given target or entity. In this paper, we perform supervised learning to perform stance detection. Firstly, we extract POS bag-of-words and use them as feature to train TIMBL. Later we tune the default TIMBL settings for optimizing the performance. We also use various other features like using all words, character n-grams or extended dataset using MPQA subjectivity lexicons. We even explore tools like MALT parser for extracting depending triples and using them as features. Finally, we perform all the analysis using Scikit-learn Random forest implementation instead of TIMBL for observing the difference in predicted accuracy.**

1. INTRODUCTION

Stance detection for tweets requires to predict whether a given tweet is in FAVOR, AGAINST or NONE. We have employed computational techniques using machine learning algorithms like K-nearest neighbor and random forest method.
Let us say we have,
Tweet: Yes, do tell US, name one of her accomplishments #SemST
Target: Hillary Clinton
Stance: AGAINST

Tweet: I don't want to be appointed to an Ambassador post.
Target Hillary Clinton
Stance: NONE

As we see in the example tweets above, every tweet talks about some specific target. However, the target name may not be directly mentioned in a tweet, but still we need to recognize the target and the tweeter's stance towards it. To do this, we need to develop a model that would recognize the relevant part of the tweet and detect the stance based on the usage of words. For example, if a tweet talks more about women power, we can predict that the author is more likely to be in favor of Hillary.
Stance detection is different than sentiment analysis, as sentiment analysis can predict if the tweet is positive, negative or neutral in general. But in stance detection, systems are to determine author's favorability towards a given target and the target may or may not be explicitly mentioned in the text.
The stance detection can have many applications like information retrieval, market research, text summarization.

2. DATASET

We have employed the method of supervised machine learning for stance detection. For this we have a training set of 2911 manually annotated tweets and a set of 1957 test tweets for testing the created model. We have data collected towards 5 different sensitive targets i.e Atheism, Feminist Movement, Hillary Clinton, Legalization of Abortion, Climate Change is a Real Concern. Each tweet has the correct stance label associated to it. We also stemmed the words for improving the accuracy of predictions.

3. DIFFERENT APPROACHES

We used TIMBL's K- nearest neighbor algorithm for constructing the hypothesis for every target. The model contained of a list of features appended by the correct label. We developed various strategies for constructing the feature vector like below:

i. POS Bag-of-words like Noun, Adjectives and verbs.
ii. Bag-of-words consisting of all words in the tweet.
iii. Weighed features based on polarity of words using MPQA Subjectivity lexicons
iv. Adding 7-grams to the POS Bag-of-words.
v. Using dependency triples generated by MALT parser as features.

We also tuned the default parameters of TIMBL to see the change in accuracy. In the next section, we have discussed the various observations and results of our experiments.

3.1. POS BAG-OF-WORDS AS FEATURES

In this strategy, we created the feature vector by extracting all the nouns, verbs and adjectives from the training data and later using them as features in our model.
For example: -
Train Tweet: Hillary is Best, Favor
BOW: Hillary, Best
Test Tweet1: 1,1, Favor
Test Tweet2: 1,0, NONE

1. We wrote a python script that would generate the feature vectors and append the correct labels to it. This was stored in separate files for every target.

2. We did the step (1) for both train.csv and test.csv.
3. We constructed the bag-of-words with only the words that were POS tagged as Adjectives: JJ, JJR, JJS , Nouns: NN,NNS,NNP or Verbs: VB,VBD,VBG,VBN,VBP,VBZ.
4. For this, we performed the POS tagging using NLTK's Penn-tree bank.

Timbl is an open-sourced memory based machine learning tool. All implemented algorithms have in common that they store some representation of the training set explicitly in memory. During testing, new cases are classified by extrapolation from the most similar stored cases.

3.2. CLASSIFICIATION USING DEFAULT SETTING

We passed our hypothesis model to the TIMBL using the basic command as:

$timbl -f Atheism_Train.txt -t Atheism_test.txt -N 10000

Here Atheism_Train.txt and Atheism_test.txt are files having lines with feature vector and attached correct label.

TIMBL returns an output with the accuracy of results and generates an output file with the feature vectors, correct labels appended with predicted label by TIMBL.

Target	Accuracy %
Atheism	65.00
Hillary Clinton	57.96
Climate Change is a Real Concern	50.88
Legalization of Abortion	58.21
Feminist Movement	57.19

Table 1: Accuracy for various targets using features with POS BOW

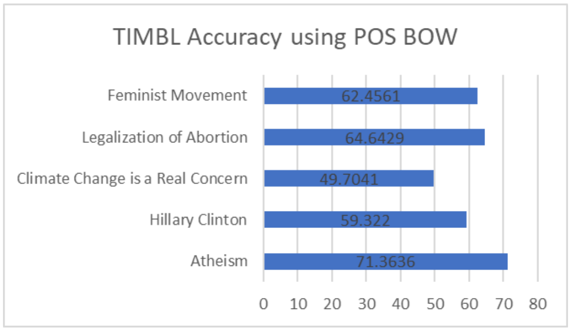

Figure 1: Graph for accuracy for various targets using features with POS BOW

In general, the TIMBL looks to perform reasonably well even on the default settings having K=1 ranging from 50% to 65%. This is understandable because the nouns, adjectives and verbs are most important parts in any sentence. So, the constructed feature vectors carry the major piece of information and helps to recognize the associated connotation effectively.

3.3. TIMBL PARAMETER OPTIMIZATION

Timbl provides the feature of passing the value of K using the command line argument. We set various different values of K to observe the change in performance.

Command Used: $timbl -f Atheism_Train.txt -t Atheism_test.txt -N 10000 -k < k value>

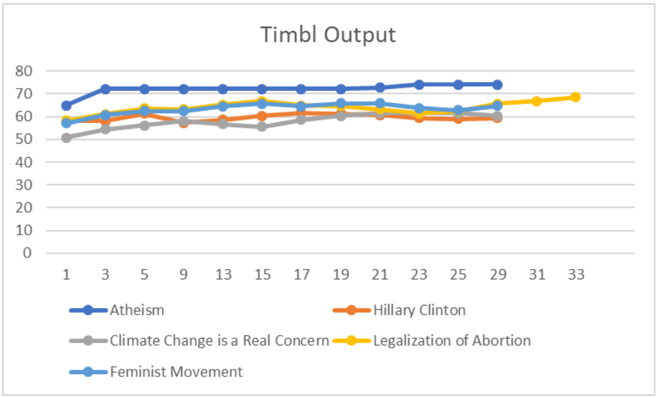

Figure 2: Graph for accuracy comparision of TIMBL output using different K-values

Here we observe that the performance is increased when the value of the default K=1 is changed. We can notice the performance to be increased up to a specific value of K and later decrease slightly and finally remains almost constant. This may be because at K=1, there are high chances of overfitting the data resulting in bad generalization of the behavior. Hence this can be taken care by manually setting the K-values. For example: At K=1 we get an accuracy of 65% for Atheism. However, it increases by 5% when the value of K is set to 3.

3.4. ALL-WORDS AS FEATURES

In this approach, we used all the words present in the training set as the BOW. During the testing, we check if a word is present in the tweet and assign 1 or 0 accordingly.

For example:

Train Tweet: Hillary is Best, Favor
BOW: Hillary, is, best
Test Tweet1: 0,0,1, None
Test Tweet2: 1,1,1, Favor

We observed the following outcome:

Target	Accuracy %
Atheism	61.81
Hillary Clinton	58.98
Climate Change is a Real Concern	53.84
Feminist Movement	57.19

Legalization of Abortion	52.14

Table 2: Accuracy for various targets using features with all-words BOW

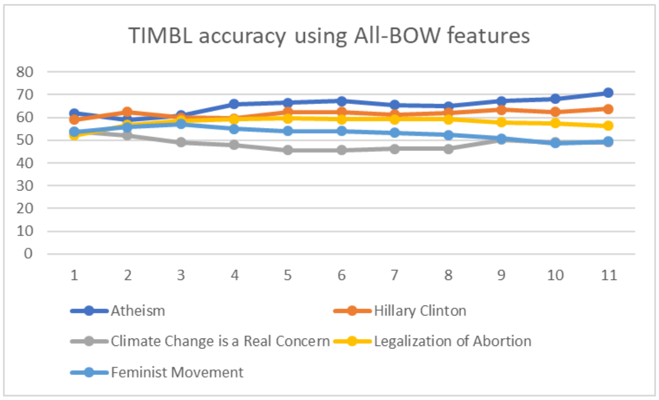

Figure 3: Graph for accuracy comparision of TIMBL output using different K-values with All-words BOW as features.

We notice a drop in the performance for most of the targets. For example, we get an accuracy of 65% for Atheism when we used POS bag of words. But when we used All-Bag-of-words instead, we observe a 4% decrease in the accuracy. This may be because the words other that nouns, adjectives and adverbs do not contribute much to the connotation of a sentence. This unnecessarily increases the size of feature vectors that causes the KNN to fall apart.

3.5. USING MPQA SUBJECTIVITY LEXICONS AS FEATURES

We have a lexicon file which has a dictionary of words and polarity attached to them. We use this lexicon file to extract features for our tweets.
Steps:

1. We look up each word in lexicon file, if it is present in a given tweet. We have also done stemming of word in tweet using nltk stemmer. This increases the chances of finding a word in lexicon file and hence help us to create better feature file.
2. If the word is present in tweet, we determine the polarity of that word from file.
3. We assign +1 for positive polarity, -1 for negative polarity and 0 for neutral polarity.
4. If a word is not found, we assign 0 i.e. it does not contribute to positive or negative stance.
5. We perform these steps for all tweets in training and test file.

We again used Timbl to train and test these feature files.
We observed that using MPQA increased our performance for all the targets. Features created using MPQA lexicon file uses word by word polarity information to determine overall stance. Thus, it gives a good performance.

Target	Accuracy %
Atheism	66.36
Hillary Clinton	54.42
Climate Change is a Real Concern	65.68
Legalization of Abortion	61.07
Feminist Movement	54.93

Table 3: Accuracy for various targets using features with extended dataset using MPQA.

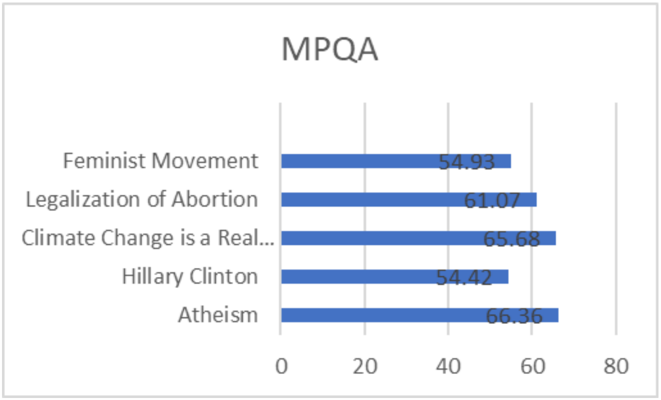

Figure 4: Graph for accuracy comparision of TIMBL output using MPQA Subjectivity lexicons.

3.6. USING CHARACTER N-GRAMS TO POS BAG OF WORDS AS FEATURES

The earlier analysis in section 3.1 showed Timbl results for features that were nouns, verbs and adjectives. In this section, we append the feature vector with character 7-grams. For this, we extracted character 7-grams from the training data and appended them to the features set.
To do this we used nltk's ngrams function in our python script.

Target	Accuracy %
Atheism	71.36
Hillary Clinton	59.32
Climate Change is a Real Concern	49.70
Legalization of Abortion	64.64
Feminist Movement	62.45

Table 4: Accuracy for various targets using features POS BOW appended with character 7-grams.

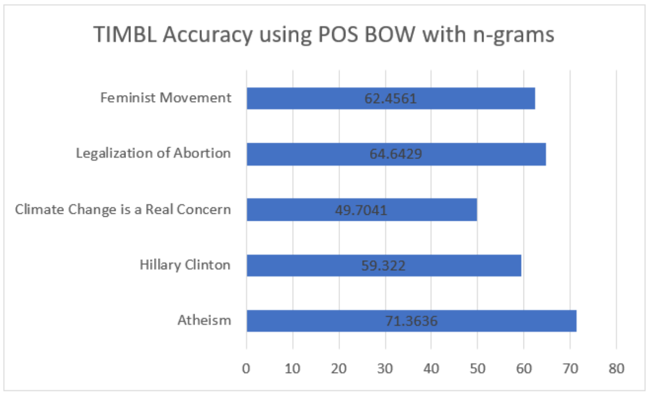

Figure 5: Graph for accuracy comparision of TIMBL output using character n-grams.

Using the n-grams strategy, we observe the maximum accuracy in our results. We can see the performance in range of 49.7%(Climate change) to 71.37%(Atheism). This is since n-grams contributes the sequential information which helps in detecting the sense of a sentence.

3.7. CONSTRUCTING FEATURES USING MALT PARSER

MaltParser is a system for data-driven dependency parsing, which can be used to induce a parsing model from treebank data and to parse new data using an induced model.[4]
The maltparser requires the input files to be in conll-x format.
We first converted the training and testing file for every target into corrresponding conll format. We did this by writing a python script that used nltk's taggedsent_to_conll function for this conversion.
We then ran the malt parser for generating the dependency triplets and used them as features.We also wrote conllx.xml to help maltparser understand the conll syntax.
Finally, We passed the training and testing file to TIMBL for running the machine learninng algorithm and obtained the results as below:

Target	Accuracy %
Atheism	59.09
Hillary Clinton	52.89
Climate Change is a Real Concern	42.33
Legalization of Abortion	53.01
Feminist Movement	51.39

Table 5: Accuracy for various targets using features extracted from MALT parser.

After comparing the accuracy given by TIMBL using dependency triples as features, we do not see any significant improvement in the performance.

3.8. PERFORMANCE COMPARISION FOR ALL STRATEGIES

In this section, we compare the results obtained by using different strategies while creating the feature vector.

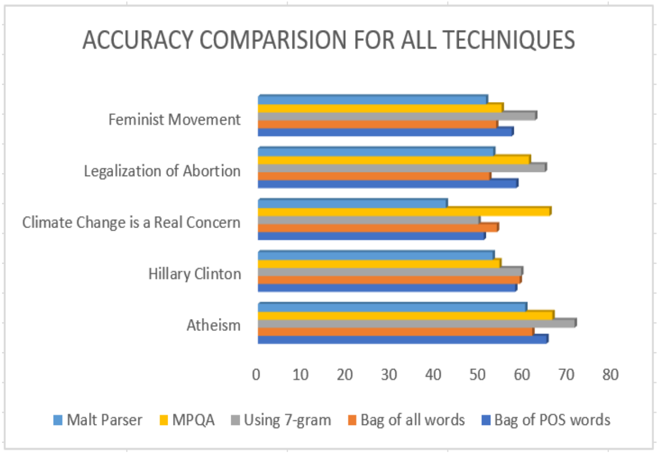

Figure 6: Comparison chart for accuracy obtained using various strategies.

After analyzing the above graph plot, we can conclude that when a bag of POS words is used in combination with character n-grams, we get the best accuracy amongst all the techniques.

4. USING SCIKIT-LEARN PYTHON

Scikit-learn is an open source machine learning package in python. It contains simple and efficient tools for data mining and data analysis. It has implementations of various supervised and unsupervised machine learning algorithms like k-means, Random forest etc. The library is built upon SciPy. We need to install SciPy before using sckit learn.

5.1. SCIKIT LEARN WITH RANDOM FOREST ALGORITHM

Random forest is a very popular machine learning algorithm for classification and regression. We combine various decision trees to create a random forest. The biggest advantage of random forest is that it never overfits the training data and it is very fast.
Earlier in this paper, we used timbl i.e. k neighbors machine learning algorithm for our experiments. Now, we will random forest and compare the accuracies with the previous results.
We are going to use random forest scikit-learn for all the below strategies:

1. Bag of words
2. POS bag of words
3. MPQA lexicon
4. N-grams
5. MALT parser output

5.2. SCIKIT RF WITH ALL BAG-OF-WORDS AS FEATURES

We see an almost similar performance when we use random forest. There is an increase in performance for Atheism, Climate Change is a Real Concern, Legalization of Abortion and Hillary Clinton targets. However, we observed a drop in performance for Feminist Movement.

TARGET	N=10	N = 100	N=50	N=200
Atheism	63.63	66.36	70.45	67.27
Climate Change i	64.49	66.86	63.31	66.86
Feminist Movement	51.58	50.87	51.93	54.38
Hillary Clinton	58.98	61.69	63.39	61.35
Legalization of Abortion	57.14	57.85	55.35	55.71

Table 6: Accuracy comparision using diferrent parameters of Scikit Random forest.

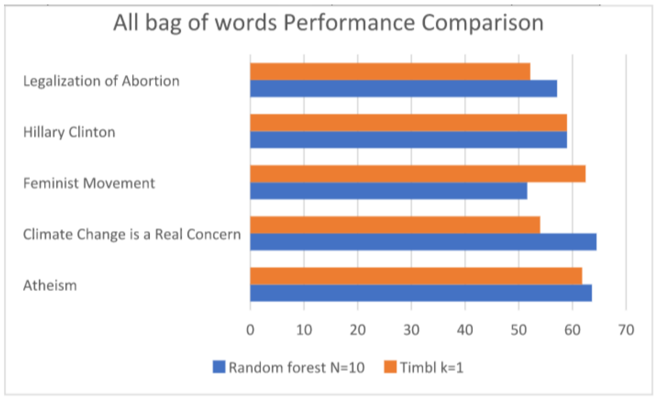

Figure 7: Figure with accuracy comparision between TIMBL vs RF.

5.3. SCIKIT RF WITH POS BAG-OF-WORDS AS FEATURES

For POS Tag feature vectors also, we observed a similar performance using random forest. It is slightly better or slightly lower than timbl.

TARGET	N=10	N = 100	N=50	N=200
Atheism	67.44	69.44	68.13	69.86
Climate Change i	60.95	63.9	63.9	64.49
Feminist Movement	49.82	54.03	54.39	55.08
Hillary Clinton	60.33	61.35	61.35	62.03
Legalization of Abortion	55.36	57.86	57.5	58.57

Table 7: Accuracy comparision using different parameters of Scikit Random forest.

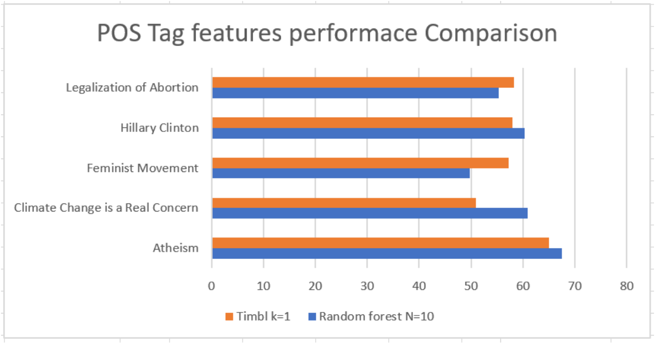

Figure 8: Figure with accuracy comparision between TIMBL vs RF.

5.4. SCIKIT RF WITH MPQA SUBJECTIVITY LEXICONS

Using MPQA Subjectivity lexicon file as feature list, we get a similar performance or increase in performance than k=1 setting for timbl.

TARGET	N=10	N = 100	N=50	N=200
Atheism	67.72	69.09	70	69.09
Climate Change i	65.68	64.49	66.27	64.49
Feminist Movement	53.68	54.03	52.98	54.38
Hillary Clinton	54.24	52.88	53.89	54.57
Legalization of Abortion	58.57	61.07	61.43	60.71

Table 8: Accuracy comparision using different parameters of Scikit Random forest.

We can observe the Comparison of results with timbl k=1 and random forest default setting in Table below.

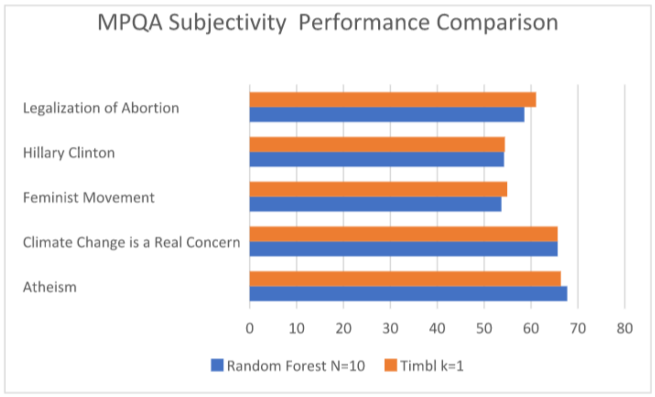

Figure 9: Figure with accuracy comparision between TIMBL vs RF.

5.5. SCIKIT RF WITH N GRAMS AS FEATURES

Using N-grams as feature vector, we observe a huge increase in performance for Climate Change is a real concern target. For Hilary Clinton also there was a great increase in performance from 59% to 69%. For other targets, we observed a decrease in performance by using random forest on default setting. We can tune in various parameters of random forest which will increase our performance.Here are the results for different setting for random forest. We changed the number of decision trees and noted the performance.

TARGET	N=10	N = 100	N=50	N=200
Atheism	71.36	75.45	74.54	75
Climate Change	70.41	68.04	64.49	68.05
Feminist Movement	58.24	52.98	53.68	54.73
Hillary Clinton	69.15	68.13	67.79	68.47
Legalization of Abortion	55	58.21	61.43	58.57

Table 9: Accuracy comparision using different parameters of Scikit Random forest.

Comparison Chart for Timbl default setting vs Random forest default setting.

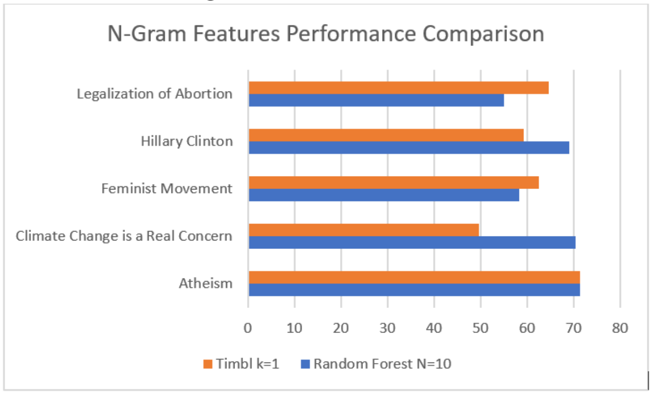

Figure 10: Figure with accuracy comparision between TIMBL vs RF.

5.6. SCIKIT RF WITH MALT OUTPUT AS FEATURES

In this experiment, we use the same training and test file with the obtained dependency triplets, as explained in the section 3.7. We observed the following results:

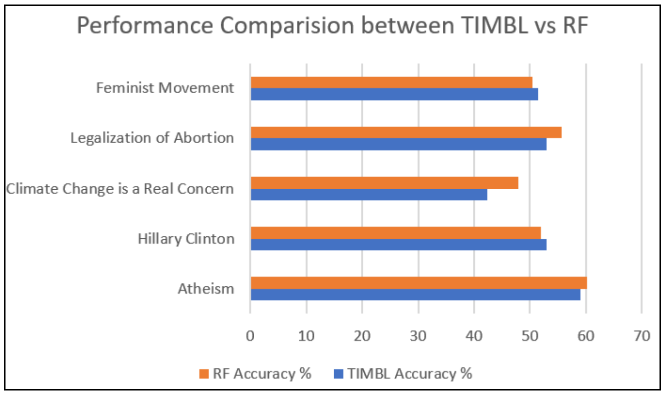

Figure 11: Figure with accuracy comparision between TIMBL vs RF for MALT Parser.

We observe that RF algorithm gives slightly better performance than TIMBL. However, the results using dependency triples are still not very impressive.

6. CONCLUSION

We employed machine learning algorithms for detecting stance of 5 targets: Hillary Clinton, Atheism, Climate change is a real concern, Feminist Movement and Legalization of Abortion. We used various Computational Linguistics Strategies for creating feature files like Bag of words, Bag of POS words, MPQA Subjectivity lexicon, N-grams and using dependency chains. We used Timbl and Scikit-learn Random forest implementation to detect stance for these 5 targets and compared the accuracies for all strategies using different machine learning implementations.

Using POS bag of words approach on Timbl gives a fairly good performance ranging from 50 to 65% for different targets. When we try to use all Bag of words, we notice a drop-in performance. This is because nouns, adjectives and verbs are the most important parts of any sentence to detect stance. We also tried and tuned using various parameter settings for Timbl by changing k value. We observed that as we increase value of k, it gives a better accuracy. After a value of k, performance either remains constant or starts dropping down. Using MPQA subjectivity lexicon increases our performance which ranges from 54% to 66% on default settings of Timbl. We also tried using N-grams to create feature vectors. It gives a relatively high performance using N-grams. When we used dependency chains approach, accuracy remains almost same

We tried to run all the feature files using Scikit-learn Random forest implementation. It slightly increased the performance in most of the cases. The results were almost like Timbl on default setting of 10 decision trees. We tried to tune in different number of decision trees values to see if we get better results. We then observed that the values were slightly higher than Timbl by tuning in different settings.

7. ACKNOWLEDGMENT

The success and outcome of this project required a lot of guidance and assistance from many people and we are extremely fortunate to have got all this during our project work. We would like to express our sincere gratitude to Prof. Sandra Keubler for providing us all support and guidance which helped us complete the project on time. We would also like to appreciate the efforts taken by our class members by presenting their brilliant ideas during group presentations and helping us out with best of their abilities.

8. REFERENCES

[1] Saif M. Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. Semeval-2016 Task 6: Detecting Stance in Tweets. June 2016. In Proceedings of the International Workshop on Semantic Evaluation (SemEval '16), San Diego, California.

[2] Peter Krejzl, Barbora Hourová, Josef Steinberger: Stance detection in online discussions

[3] Parinaz Sobhani, Saif M. Mohammad, and Svetlana Kiritchenko. Detecting Stance in Tweets And Analyzing its Interaction with Sentiment. August 2016. In Proceedings of the Joint Conference on Lexical and Computational Semantics (*Sem), Berlin, Germany.

[4] http://www.maltparser.org/userguide.html: Official website of MaltParser tool.

[5] http://scikit-learn.org/stable/modules/ensemble.html#forests-of-randomized-trees : Introduction to Scikit-learn Random forest package

[6] Can Liu, Wen Li, Bradford Demarest, Yue Chen, Sara Couture, Daniel Dakota, Nikita Haduong, Noah Kaufman, Andrew Lamont, Manan Pancholi, Kenneth Steimel, Sandra Kubler : IUCL at SemEval-2016 Task 6: An Ensemble Model for Stance Detection in Twitter.