Jurnal
Teknologi

Full Paper

A COMPARATIVE STUDY OF STATISTICAL AND NATURAL LANGUAGE PROCESSING TECHNIQUES FOR SENTIMENT ANALYSIS

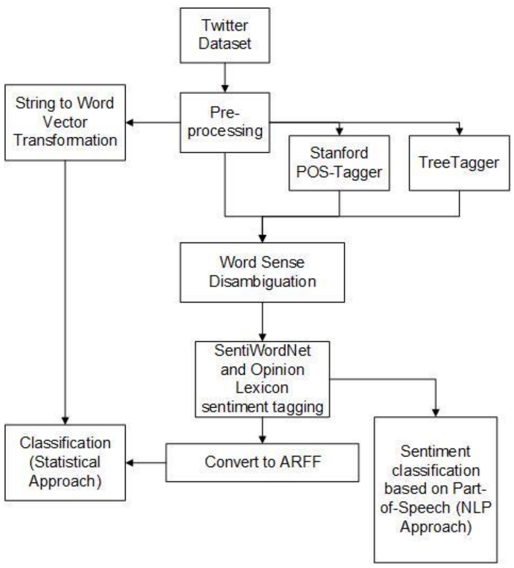
Wai-Howe Khong*, Lay-Ki Soon, Hui-Ngo Goh

Faculty of Computer Informatics, Multimedia University, Malaysia

Article history
Received
15 May 2015
Received in revised form
1 July 2015
Accepted
11 August 2015

*Corresponding author
1141600049@student.mmu.edu.my

Graphical abstract



Abstract

Sentiment analysis has emerged as one of the most powerful tools in business intelligence. With the aim of proposing an effective sentiment analysis technique, we have performed experiments on analyzing the sentiments of 3,424 tweets using both statistical and natural language processing (NLP) techniques as part of our background study. For statistical technique, machine learning algorithms such as Support Vector Machines (SVMs), decision trees and Naïve Bayes have been explored. The results show that SVM consistently outperformed the rest in both classifications. As for sentiment analysis using NLP techniques, we used two different tagging methods for part-of-speech (POS) tagging. Subsequently, the output is used for word sense disambiguation (WSD) using WordNet, followed by sentiment identification using SentiWordNet. Our experimental results indicate that adjectives and adverbs are sufficient to infer the sentiment of tweets compared to other combinations. Comparatively, the statistical approach records higher accuracy than the NLP approach by approximately 17%.

Keywords: Natural language processing, sentiment analysis, word sense disambiguation

1.0 INTRODUCTION

Internet is getting a wider audience, so is the number of opinions posted on the net, especially on social media. Amongst all these social media sites, Twitter is one of the social networking site that the user can easily express their opinion and reviews rapid-ly. Twitter is also known as a microblog because it allows users to read and post a short 140-character messages. Over time, Twitter has become a fertile source of product reviews, which has become one of the primary focuses for many sentiment analysis related research.

Sentiment analysis has been approached by using two main types of datasets. One is the datasets with two-class labels of positive and negative as the sentiment polarity, while the other one is three-class labels that includes neutral as the sentiment polarity. Neutral samples can be taken into account as it will enhance the accuracy of the classification (Koppel *et al*., 2005) [1]. Statistical approach using machine learning algorithms have been used to analyse the datasets, which includes Naïve Bayes, Maximum Entropy, Support Vector Machines (SVM), and k-Nearest Neighbours algorithm (KNN) (Pang *et al*., 2002 [2], Jin *et al*., 2012 [3]). As for natural language processing (NLP) approach, Stanford POS Tagger and TreeTagger have been widely used for part of speech tagging, while Lesk algorithm and WordNet have been used for Word Sense Disambiguation (WSD) (Pedersen *et al*., 2005) [4].

In this paper, we present our experiments on sentiment analysis using both statistical method and NLP techniques and subsequently compare the results. For the purpose of the experiments, we have adopted Twitter dataset which have been labelled as positive, negative, neutral and irrelevant. Tweets that are tagged as “Irrelevant” will not be taken into account and we attempted the experiments using both three-class labels and two-class labels tweets. Two-class labels consist of positive and negative tweets while three-class labels include neutral tweets.

For NLP approach, two different tagging methods were applied on the tweets, where the output was then sent for WSD using Lesk Algorithm and WordNet. Next, word level sentiment identification is performed by using SentiWordNet (Baccianella *et al*., 2008) [5]. In order to determine the sentiment polarity of a tweet, the sentiments of words with different combination part-of-speech (POS) tagging within the tweet were investigated. On the other hand, classifiers built using SVM, Naïve Bayes and Decision Tree were explored for the statistical approach. The tweets are represented using different features, such as word and term frequency-inverse document frequency (TF-IDF). In order to investigate the impact of NLP in statistical sentiment analysis, classifiers were also applied on datasets that have been processed by NLP. Section 3 and 4 detail the experiments.

For the remaining of this paper, Section 2 briefly discusses the related word, Section 3 presents the methodology of our experiments and Section 4 outlines the experimental dataset. Next, Section 5 presents the experimental results and lastly the paper is concluded in Section 6.

2.0 RELATED WORK

Sentiment analysis has been applied in two main different ways namely the statistical approach (Jin *et al*., 2012 [2], Pang *et al*., 2002 [3]) using Naïve Bayes, Maximum Entropy, Support Vector Machines (SVM), and k-Nearest Neighbours algorithm (KNN) as well as the natural language processing (NLP) approach (Passonneau, 2011) [6]. The neutral samples has been considered as it enhances the accuracy of the classification (Koppel *et al*., 2005 [1]). It was mentioned (Ku *et al*., 2006 [7]) that machine learning algorithm is not suitable for word level opinion extraction (Pang *et al*. 2002 [3]). However, our experiment results show that the difference is quite small. A combination of randomwalk algorithm that weights synsets from the text with polarity scores provided by SentiWordNet (Montejo-Ráez *et al*., 2014) [8] has been approached too as it does not suffer from the disadvantages associated with supervised methods.

3.0 METHODOLOGY

In this section, we will describe the process from data pre-processing to classification as shown in Figure 1.

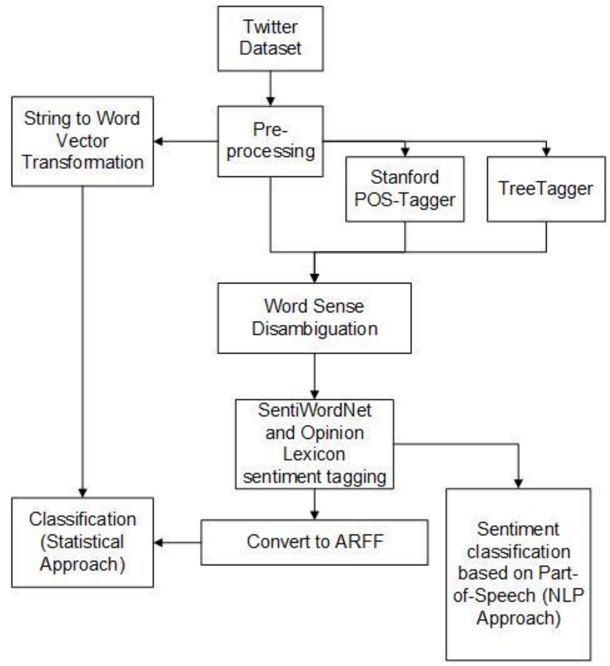


Figure 1 Visual representation of Section 3

3.1 Data Pre-processing

The twitter dataset that we acquired has been benchmarked. Before the cleaning, tweets that are tagged as irrelevant will be removed. During data preprocessing, tweets will be manually processed to eliminate most of the noise that present in the tweets, from emoticons, links to short forms. This step is required to produce a cleaner version of the corpora for more accurate sentiment analysis. The details below are the steps taken to clean the tweet:

Convert HTML codes and Unicode: There are some symbols in these tweets that uses HTML codes like “>” that refers to “>”, “<” that refers to “<”, and Unicode like “\u2018” which means single open quote. These HTML codes and Unicode were replaced accordingly. This step is necessary for us to convert emoticons into words.

Links removal: Any links or URLs which starts with “http” were removed from the tweets. Replacing it with a representative token however will increase unnecessary frequency in a single token.

Removal of hashtag and topic indication sign: Hashtags and topics will not be re-moved as they pose important part-of-speech for later analysis. Instead, only the “@” and “#” signs which represents the topic and hashtag respectively were remove, for example “#IOS, @Microsoft” become “IOS, Microsoft”.

Emoticon conversion: Emoticons were converted to what they mean in word form. For example, “:)” was converted to “happy”, “T_T” was converted to “cry”. There are altogether 186 emoticons converted.

Abbreviation conversion: Abbreviations will be converted into their full words, for example “rofl” will be changed into "Roll on floor laughing". There are a total of 388 sets of abbreviations used in these conversion.

Colon removal: The colon symbol in tweets will be removed due to some bugs with the WSD tool that is unable to completely process tweets that contain colon.

Removal of repeating characters: Tweets with words containing repeated characters were identified. For example, "YAAAAAAAAY". These words will be mapped to their base form, for example "yay" for the final cleaning. This is to prevent accidental conversion of correct words with more than two same characters occurring side by side in a word, furthermore, this will prevent conversion of links with "www" in it.

3.2 Word Vector Filtering

In this section, the pre-processed dataset from section 3.1 was converted to two different datasets. The datasets consists of three-class labels, including tweets with positive, negative and neutral samples while the two-class labels consists of positive and negative tweets only. The word datasets were then sent to Waikato Environment for Knowledge Analysis (WEKA) for string to word vector transformation. Inspired by (Huang *et al*., 2012) [3], we have also used different representation for the features in these two datasets during the word vector transformation:

BOOL: Boolean representation where 1 is used if a word appears in the tweet and 0 for none.

TF-IDF (Term Frequency Inverse Document Frequency): Frequency of a word occurs in a tweet and the measure of common importance of the words within a tweet across all tweets in the datasets

BOOL-IDF (Boolean Inverse Document Frequency): Boolean representation and the measure of common importance of the words within a tweet across all tweets in the datasets.

Next, we have also converted the datasets into two formats, one with lower case tokens and word stemming using Snowball stemmer, and the other one without these two further processing steps. The lower case token is to test if it enhance the classification done in section 3.7 as some words consist of both casing, for example: "Google" and "google" which were the same word but being separated into 2 different attributes. The output of this step is used for statistical sentiment analysis experiment. At the end of this step, six ARFF files will be created, as shown in Table 1.

Table 1 List of output from section 3.2

No.	Format of Words (features)
1	BOOL
2	TF-IDF
3	BOOL-IDF
4	BOOL with lower case and snowball stemmer
5	TF-IDF with lower case and snowball stemmer
6	BOOL-IDF with lower case and snowball stemmer

3.3 Part-of-Speech Tagging

For part-of-speech tagging, we used the processed datasets produced from the pre-processing in section 3.1. In this step, the dataset were divided into two dataset, one with mixed-case tokens (words) while the other with lower case tokens. Next, both datasets will be tagged using two POS tagging tools:

TreeTagger[1]: Tree Tagger is a tool for annotating text with part-of-speech and lemma information. It was developed by Helmut Schmid in the TC project at the Institute for Computational Linguistics of the University of Stuttgart.

Stanford Log-linear Part-Of-Speech Tagger[2]: For processing English language input, Stanford POS Tagger uses Penn Treebank English POS tag set.

At the end of this step, four datasets were produced. The first one containing POS-tagged dataset in lower case while the second one containing POS-tagged in mixed-case using TreeTagger. The third and fourth POS-tagged datasets are in lower case and mixed-case using Stanford POS respectively.

3.4 Word Sense Disambiguation and SentiWordNet

After section 3.3, the four POS-tagged datasets will go through Word Sense Disambiguation (WSD) using WordNet::SenseRelate::AllWords[3] . WSD is the process of identifying the sense of a polysemic word. It usually uses WordNet as a reference sense inventory for English, which is a computational lexicon that encodes concepts as synonym sets. Every word after WSD will be changed from "living" to "liv-ing#a#3", where "#a" is the POS and "#3" is the word sense. Some words that do not contain word sense or not in WordNet but contains POS will be changed to "living#a" instead. If the POS and word sense of a word is undetermined, it will remain unchanged.

The outputs of WSD dataset were then sent to SentiWordNet (Baccianella *et al*., 2010) [5], a lexical resource based on the well-known WordNet for opinion mining. SentiWordNet assigns to each synset of WordNet, the basic item of information in WordNet and it represents a "concept" that is unambiguous, three sentiment scores: positivity, negativity, objectivity.

For the weighted sentiment on SentiWordNet, we have taken into account the sentiment with the largest weight and with the matched POS attached to the word sense. For example, "a 0.5 0.125 living#3", indicates that "a" is the part-of-speech of the word (living), "0.5" is the positive weight and "0.125" is the negative weight and "living#3" is the word sense. Since "0.5" is more than "0.125", the word will be considered as positive and vice versa. In case of same weight, it will be tagged as neutral. For the word however, it will be changed from "living#3"

[1] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/
[2] http://nlp.stanford.edu/software/tagger.shtml
[3] http://maraca.d.umn.edu/allwords/allwords.html

to “living#a#3” where “a” is the POS adverb of this word to match the output after WSD.
For words that are not included in SentiWordNet, they will be checked against a compiled list of opinion lexicon[1] (Hu *et al*., 2004) to determine the sentiment polarity of a word. The words were then tagged as p, g or n respectively, where p represent positive, g represents negative and n represents neutral

3.5 ARFF File with Features from NLP output

In this section, datasets processed from section 3.4 will be converted into Attribute-Relation File Format (ARFF) in order to be processed by WEKA. The benchmarked dataset’s sentiment polarity will be added back to each tweet for the training purpose. Each word will be a single attribute with three possible
value, {p,g,n} (e.g.: @attribute great {p,g,n}) that was tagged from the previous section. Two sets of output will be produced, one with neutral tweets (labelled during data acquisition) and another without the neutral tweets.

3.6 Sentiment Classification using Word Level Sentiment from Different Part-of-Speech Combination

In this section, we will take the output from section 3.4 which has gone through section 3.3 so that we are able to extract a combination of POS with sentiment. We integrated a different combinations of adverbs, adjectives, nouns and verbs in a tweet with sentiment to derive the general sentiment of the tweet. Section 5 discusses and compares the results of experiments performed on the output of this section. Table 2 shows the output of cleaned sample tweet, different POS-tagged, and their WSD and sentiment identification sample output.

3.7 Classification

The result sets from section 3.2 and 3.5 will be used for classification with these three classifiers:
Naive Bayes: Naive Bayes is a simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features [10]. Despite its simplicity, Naive Bayes can often outperform more sophisticated classification methods.
Support Vector Machines (SVM): SVM is primarily a classier method that performs classification tasks by constructing hyperplanes in a multidimensional space that separates cases of different class labels. SVM supports both regression and classification tasks and can handle multiple continuous and categorical variables [11].
Decision Tree Classifier: Decision Tree Classifier is a simple and widely used classification technique. It applies a straightforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receive an answer, a follow-up question is asked until a conclusion about the class label of the record is reach [12].

These algorithms were chosen for the purpose of our experiments as they were highly accurate as reported by most research work (Jin *et al*., 2012 [3], Ku *et al*., 2006 [7]). We used 10-fold cross validation is all the experiments in WEKA.

4.0 EXPERIMENTAL DATASET

The Twitter dataset that we downloaded consists of 5,113 tweets. Amongst those tweets, 1,689 of them are being labelled as “irrelevant”. For example, some irrelevant tweets consists of foreign languages, URL only or Unicode characters that appears as “???”. All these tweets will not be included in any experiment, which left us with 3,424 tweets with sentiment of either positive, negative or neutral. The tweets focus on four main topics, namely Apple, Microsoft, Google and Twitter itself. Table 3 shows a breakdown of the tweets. The sentiment of the dataset were manually labelled.

Table 2 Sample tweet and process from POS-Tagging to sentiment identification

Process	Tweet
Original Tweet	RT @MN2NOVA: Love ios5 Easter eggs. Pull down from middle top to bottom and see what pulls down. Awesome little feature! #ios5 @apple
After Section 3.1	retweet MN2NOVA Love ios5 Easter eggs. Pull down from middle top to bottom and see what pulls down. Awesome little feature! ios5 apple
POS Tagger	retweet/VB MN2NOVA/NNP Love/NNP ios5/NNP Easter/NNP eggs/NNS ./. Pull/VB down/RB from/IN middle/JJ top/NN to/TO bottom/NN and/CC see/VB what/WP pulls/VBZ down/RP ./. Awesome/JJ little/JJ feature/NN !/. Ios5/NN apple/NN
After WSD (POS Tagger)	retweet#v MN2NOVA#n Love#n#1 ios5#n Easter#n#1 egg#n#3 Pull#v#1 down#r#1 from#r middle#a#1 top#n#4 to bottom#n#4 and see#v#5 what pull#v#1 down#r#1 Awesome#a#1 little#a#1 feature#n#1 ios5#n apple#n#2
After Sentiment identification with SentiWordNet and opinion lexicon (POS Tagger)	retweet#n mn2nova#n love#p ios5#n easter#g egg#n pull#n down#g from#n middle#n top#n to#n bottom#n and#n see#p what#n pull#n down#g awesome#p little#g feature#n ios5#n apple#n

[1] http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

TreeTagger	retweet/NP MN2NOVA/NP Love/NP ios5/NP Easter/NP eggs/NNS ./SENT Pull/VV down/RB from/IN middle/JJ top/NN to/TO bottom/VV and/CC see/VV what/WP pulls/VVZ down/RP ./SENT Awesome/JJ little/JJ feature/NN !/SENT ios5/JJ apple/NN
After WSD (TreeTagger)	retweet MN2NOVA Love ios5 Easter egg#n#3 . Pull down#r#1 from#r middle#a#1 top#n#9 to bottom and see what pulls down#r#2 . Awesome#a#1 little#a#1 feature#n#1 ios5#a apple#n#2
After Sentiment identification with SentiWordNet and opinion lexicon (TreeTagger)	retweet#n mn2nova#n love#p ios5#n easter#n egg#n .#n pull#n down#n from#n middle#n top#n to#n bottom#n and#n see#n what#n pulls#n down#n .#n awesome#p little#g feature#n ios5#n apple#n

Table 3 Detailed breakdown of the dataset

Total Tweets	5113
Apple positive tweets	164
Google positive tweets	202
Microsoft positive tweets	91
Twitter positive tweets	62
Total Positive Tweets	519
Apple negative tweets	316
Google negative tweets	57
Microsoft negative tweets	132
Twitter negative tweets	67
Total Negative Tweets	572
Apple neutral tweets	523
Google neutral tweets	579
Microsoft neutral tweets	641
Twitter neutral tweets	590
Total Neutral Tweets	2333
Total Irrelevant Tweets	1689
Total Tweets after removing Irrelevant Tweets	**3424**

5.0 EXPERIMENTAL RESULTS

5.1 Results from NLP Approach

Upon completing all the sections till Section 3.6, the tweets were POS-tagged, word sense disambiguated and finally sentiment labelled. In this experiment, we examined the accuracy of sentiment analysis by examining different combination of words with sentiment based on their POS. We first checked for the POS that we want to explore in a tweet, then extracted only the words that are tagged with the chosen POS. After the extraction, we compared the number of positively labelled words with negatively labelled words. If the frequency of positive labelled word is greater than negative labelled words, the tweet will be labelled as positive and vice versa. However, if the frequencies of both positive and negative labelled word are the same or none has been identified, then the tweet will be labelled as neutral instead.

Here is a combination of POS that we extracted from the tweets, where a = adjective, r = adverb, n = noun and v = verb:

ar_addnv: we first count the frequencies of adjective and adverb in a tweet with both positive and negative sentiments respectively. If the count is 0 for both, we will then look into noun labels. If it is 0 again, we would finally look into verbs. If we are able to get a count on either one, we will stop counting and will decide on the sentiment of the tweet. For example, a tweet contains 2 positive nouns and 2 negative verbs. First, we count adjective and adverb, which we will get 0 count for sentiment labelled words. Then we count again noun that have sentiment polarity and we get 2 positive count. Since we are able to get a count on noun, we will not count the sentiment labelled verbs. In the end, we get 2 positive count, which we will label this tweet as positive.

ar_addvn: same as the above but with different arrangement, if the count of both adjective and adverb is 0, we will look into verbs first then into noun.

ar: counts both adjective and adverb with sentiment polarity.

arn: counts adjective, adverb and noun with sentiment polarity .

arv: counts adjective, adverb and verb with sentiment polarity .

nv: counts both noun and verb with sentiment polarity

arnv: counts adjective, adverb, noun and verb with sentiment polarity at the same time. The difference between this and the first 2 is that all four POS with labelled sentiment will be counted at the same time, while the formal ones will look into adjective and adverb first then into a different priority of noun and verb. For example, if a tweet contains 1 positive labelled adjective, 1 negative labelled adverb, 0 noun and 2 negative labelled verbs, all these sentiment labelled POS will be counted at the same time without order. Finally the tweet will be labelled as negative as the number of negative labelled POS is larger than positive labelled POS.

Table 4 shows the results of our experiment. As observed, dataset that is represented in mixed-case together with POS tagged by TreeTagger produces the best result compared to other methods, while Stanford POS Tagger combined with lower case gives the worst result for all combinations except for *nv* (45.8820%). Besides, the combination of adjective and adverbs outperforms the other combinations as it achieves the best accuracy. Amongst the *ar* POS combination, dataset with POS tagged by TreeTagger in mixed-case produces the best result (55.4030%). Given the accuracy obtained by the different combinations of POS, we conclude that as the number of POS to be extracted from the tweets increases, the accuracy decreases.

Table 4 Accuracy of sentiment classification based on Part-of-Speech

POS type \Tagger & casing	TreeTagger Mixed Case	POS Tagger Mixed Case	TreeTagger Lower Case	POS Tagger Lower Case
ar_addnv	**48.3061%**	43.0491%	45.2979%	41.7056%
ar_addvn	**47.7804%**	42.3481%	44.7138%	41.3259%
ar	**55.4030%**	54.8773%	52.7161%	52.1028%
arn	**52.9206%**	50.6717%	49.9124%	49.4451%
arv	**49.2699%**	46.1449%	46.7874%	43.8960%
nv	**53.1834%**	45.5023%	51.2850%	**45.8820%**
arnv	**47.6051%**	43.3703%	44.9474%	41.7640%

5.2 Results from Statistical Approach

As mentioned in section 3.6, the outputs were experimented using multiple classifiers in 10 fold cross-validation. We used two variations of SVM, which is libSVM and sequential minimal optimization (SMO) classifiers. In libSVM, we used C-support vector classification (C-SVC) for two-class labels and nu-support vector classification (nu-SVC) for three-class labels. The reason why nu-SVC has been used to classify three-class label was that C-SVC only supports two-class labels. As for the decision tree classifier, we applied J48 as the default decision tree classifier. WSD-Raw output is the raw text after running through section 3.1 to clean off the noise from the output, which mean POS tagging was not applied before going into WSD. Dataset with lower case and snowball stemmer are labelled as LS, which are BOOL-IDF LS, TF-IDF LS and BOOL LS.

As observed from Figure 2, WSD-Raw with libSVM classifier produces the highest accuracy (82.4015%) while J48 for both BOOL and BOOL-IDF representations achieve the least accuracy (67.1861%). In most cases, libSVM outperforms all the other classifiers while J48 is the least accurate classifier

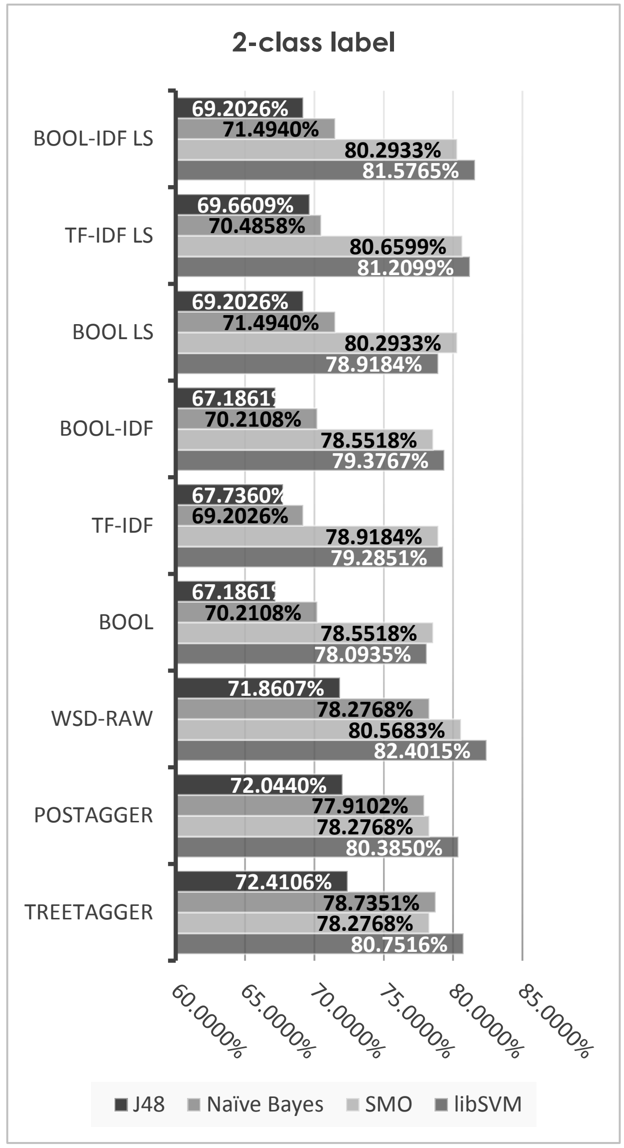


Figure 2 Two-class label classified by statistical approach

For three-class labelled datasets (Figure 3), it is interesting to note that dataset contains features represented in TF-IDF LS (TF-IDF with lower case and Snowball stemmer) achieves the best result. In terms of classification algorithms, SMO classifier is the most accurate (76.9638%). On the other hand, Naïve Bayes classifier yields the least accurate result (60.1928%). As for the all the results generated by libSVM, it yields the same result regardless of dataset representations. After a close inspection, it seems that all positive and negative tweets are being labelled as neutral tweets (68.1367%). This might be the result of neutral samples having the highest number of tweets, even with positive and negative tweets combined.

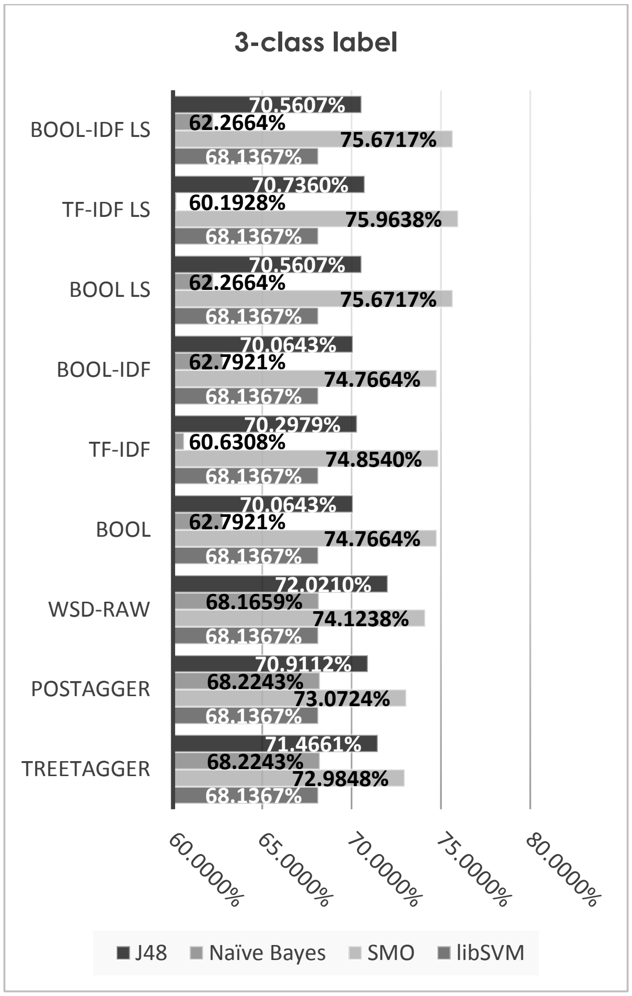


Figure 3 Three-class label classified by statistical approach

If we compare both charts, two-class labels produces slightly accurate result compared to three-class label. Moreover, both SVM classifiers seems to produce a better result across both charts. J48 classifier produces results that are quite near to each other in both two-class and three-class datasets. On the other hand, Naïve Bayes classifier performs better with datasets represented using NLP-produced features (TreeTagger-tagged, POS-tagger tagged and WSD-Raw).

6.0 CONCLUSION

In this paper, we present a new approach of combining POS tagging together with WSD and SentiWordNet to perform sentiment analysis. However, the accuracy does not shine as much as the results generated from statistical approach. From the results presented in Section 5, SVM has been proven to perform better compared to other classifiers. Contrary to other research work, the inclusion of neutral samples however does not contribute to the accuracy of the said classifier (Koppel *et al*., 2006 [1], Vryniotis, 2013 [13]). Statistical learning on dataset represented using output of NLP techniques, namely word sense after WSD and word-level sentiment identification outperforms datasets represented in word frequency and TF-IDF by a very small margin. For future work, we plan to work on experiments that extract and identify the words that targets the subject of the tweet and only extract the sentence with the subject.

References

[1] Koppel, M., & Schler, J. 2006. The Importance of Neutral Examples for Learning Sentiment. Computational Intelligence. Retrieved from http://onlinelibrary.wiley.com/doi/10.1111/j.1467-8640.2006.00276.x/abstract.

[2] B. Pang, L. Lee, H. Rd, and S. Jose. 2002. Thumbs Up ? *Sentiment Classification using Machine Learning Techniques*. July: 79-86.

[3] H. Jin, M. Huang, and X. Zhu. 2012. Sentiment Analysis with Multi-source Product Reviews. *Intell. Comput. Technol*. 301-308.

[4] T. Pedersen and S. Banerjee. 2005. Maximizing Semantic Relatedness to Perform. March.

[5] Baccianella, S., Esuli, A., & Sebastiani, F. 2008. SENTIWORDNET 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. 0: 2200-2204.S.

[6] R. Passonneau. 2011. Sentiment Analysis of Twitter Data. June: 30-38.

[7] Ku, L., Liang, Y., Chen, H., Lun-Wei, K., Yu-Ting, L., & Hsin-Hsi, C. 2006. Opinion Extraction, Summarization and Tracking in News and Blog Corpora. In *Artificial Intelligence*. http://doi.org/citeulike-article-id:2913694. 100-107.

[8] A. Montejo-Ráez, E. Martínez-Cámara, M. T. Martín-Valdivia, and L. A. Ureña-López. 2014. Ranked WordNet graph for Sentiment Polarity Classification in Twitte. *Comput. Speech Lang*. 28(1): 93-107.

[9] M. Hu, B. Liu, and S. M. Street. 2004. Mining and Summarizing Customer Reviews.

[10] Naive Bayes classifier. 2014. [Online]. Available: http://en.wikipedia.org/wiki/Naive_Bayes_classifier.

[11] Support Vector Machines (SVM). 2014. [Online]. Available: http://www.statsoft.com/textbook/support-vector-machines. [Accessed: 03-Nov-2014].

[12] Decision Tree Classifier. 2014. [Online]. Available: http://mines.humanoriented.com/classes/2010/fall/csci568/portfolio_exports/lguo/decisionTree.html. [Accessed: 03-Nov-2014].

[13] V. Vryniotis. 2013. The importance of Neutral Class in Sentiment Analysis | DatumBox. [Online]. Available: http://blog.datumbox.com/the-importance-of-neutral-class-in-sentiment-analysis/. [Accessed: 10-Apr-2014].