# Examining machine learning techniques in business news headline sentiment analysis

Seong Liang Ooi Lim, Hooi Mei Lim, Eng Kee Tan, Tien-Ping Tan*

School of Computer Sciences, Universiti Sains Malaysia, 11800 Penang, Malaysia
tienping@usm.my

**Abstract.** Sentiment analysis is a natural language processing task that attempts to predict the opinion, feeling or view of a text. The interest in sentiment analysis has been rising due to the availability of a large amount of sentiment corpus and the enormous potential of sentiment analysis applications. This work attempts to evaluate different machine learning techniques in predicting the sentiment of the readers toward business news headlines. News articles report events that have happened in the world and expert opinions. These are factors that will affect market sentiment, and a headline can be considered as a summary of an article in a single sentence. In this study, we constructed a sentiment analysis corpus which consists of business news headlines. We examined two different approaches, namely text classification and recurrent neural network (RNN) in predicting the sentiment of a business news headline. For text classification approach, multilayer perceptron (MLP) classifier, multinomial naïve Bayes, complement naïve Bayes and decision trees were experimented. On the other hand, for the RNN approach, we evaluated the typical RNN architecture and the encoder-decoder architecture in predicting the sentiment.

**Keywords:** sentimental analysis, text classification, recurrent neural networks, business/finance news.

## 1 Introduction

Sentiment analysis is a natural language processing (NLP) task that attempts to predict the opinion, feeling or view of a text. Thus, it is sometimes also referred to as opinion mining. The interest in sentiment analysis has been rising due to the availability of a large amount of sentiment corpus and the enormous potential of sentiment analysis applications, such as tracking the political inclination of the public [1], reviewing customer satisfaction toward a product or service [2], improving customer relationship management, and detecting the well-being of the people. Sentiment prediction is done by either assigning a category, such as positive, negative or neutral, or a value to a text.

Sentiment analysis can be performed at three different levels. Document-level sentiment analysis treats individual document such as a review post as a basic data unit to assign with a sentiment class. However, it does not work well with documents which contain comparisons of different items and aspects. Sentence-level sentiment analysis predicts the sentiment of sentences in a document, while aspect-/feature-/phrase-level

sentiment analysis works with even more granular data units, which are words or phrases that carry the opinion.

While many works related to sentiment analysis attempted to predict the sentiments of the authors, we are interested in analyzing the sentiments of the readers. See examples in Table 1. This work focuses on business/finance news headlines. A headline can be seen as the summary of an article in a sentence. Often, business news headlines also describe experts' opinion on a certain issue. Events that happen in the world and expert opinions are factors that will affect market sentiment. Market sentiment is the overall emotion of traders and investors toward a particular security or market. When the market sentiment is bullish/positive, stock prices may go up, and on the other hand, when the market sentiment is bearish/negative, the opposite is likely to happen. Thus, business news headlines are good indicators to predict the direction of the market.

**Table 1.** Examples of news headlines and their sentiment

| Sentence | Sentiment |
|---|---|
| 1. Palm oil may fall more into of 2,162-2,178 ringgit range | Negative |
| 2. Genting Malaysia 's growth prospects still seen positive despite us setback | Positive |
| 3. Wall Street extends rally, tech leads S&P, Nasdaq to record highs | Positive |
| 4. Ecoworld aware of women 's influence in the workplace | Neutral |

## 2 Sentimental Analysis

Sentiment analysis usually involves several essential steps, namely data collection, data pre-processing, feature extraction, feature selection, and sentiment classification. Data collection involves acquiring textual data from the relevant sources, which could be review websites, blogs, microblogs or datasets [3]. Web scraping can be used to collect these data which are later stored in the database. Some data are already annotated with sentiment information, such as text on hotel booking, product satisfaction, etc., but some collected data require manual annotation. Sentiment annotation can be carried out by assigning each text a sentiment class, which can be either positive, negative, neutral or conflict [3], or it can be assigned a value. Next, the raw data will be pre-processed using NLP techniques. Tokenization is employed to split up the texts into tokens by eliminating whitespaces and unwanted punctuations. Normalization is used to convert all the characters to lowercase or uppercase. Stemming extracts fixed parts of the words, while stop word removal removes common words such as "a", "an", "the", and etc.

Depending on the sentiment analysis approach used, feature extraction may or may not be applied. For machine learning approaches that use the bag-of-words model such as naïve Bayes, decision tree, and support vector machine (SVM), feature extraction is required. On the other hand, approaches that use artificial neural networks may not need to perform feature extraction, but word embedding. During feature extraction, potentially useful features such as term frequency, term co-occurrence such as n-grams, part of speech (POS) information, opinion words based on relevant lexicon and syntactic dependency are identified and extracted. Additional features used in [4] for aspect-

based sentiment analysis include Word-Aspect Association Lexicon which links opinion words to the aspects which they usually describe. Besides, negation words such as "not" and "never" must be considered so that words appearing in a negated context are not processed wrongly. The extracted features can be filtered through feature selection to reduce the size of the feature vector and thus enhance performance. Information gain, odd ratio, term frequency-inverse document frequency (TF-IDF) and POS can be used for feature weighting mechanisms. Moreover, ablation analysis can be conducted to find out which features contribute to the highest accuracy gains.

Once the features are obtained, sentiment modelling can be carried out using machine learning techniques. There are two types of machine learning: supervised and unsupervised. In supervised machine learning, the data must be annotated with the appropriate class. The most commonly used classifiers are naïve Bayes, SVM, and maximum entropy. Classification is done based on the selected features extracted from the text. The most commonly used features are: frequency of term, part of speech, and negation. Naïve Bayes shows better precision compared to the other classifiers [5].

On the other hand, deep learning approaches use multiple layers of processing units for modelling; lower layers learn simple features while higher layers will learn more complex features from features derived by lower layers. The most commonly used approaches in deep learning are convolutional neural network (CNN) and recurrent neural network (RNN) [5]. A neural network consists of 3 main layers, which are the input/embedding layer, hidden layer, and output layer. Word embedding layer is used to learn and capture the semantic and the relationship of words. It can be trained by using neural networks or matrix factorization. Then, the word vector will be input to the hidden layer for feature extraction and the output layer will have an activation function such as Softmax function for final classification. A very comprehensive study on deep neural networks in sentiment analysis was carried out by Zhang et al. [6].

There are also other sentiment modeling approaches that do not use machine learning technique. One approach uses a sentiment lexicon in predicting the sentiment of a text. Sentiment lexicon contains a list of words, each with a score that indicates the sentiment polarity. First, the scores of subjective words are summed up separately according to positive, negative, and neutral classes. The overall polarity of a text is determined by the class with the highest score. For example, if a text contains more positive words, then its polarity will be positive [3]. There are three methods to construct a sentiment lexicon: manual construction, dictionary-based approach, and corpus-based approach. Manual construction requires experts to manually create the lexicon which is very time-consuming. In a dictionary-based approach, first, a small set of opinion words known as the seed list is collected manually. It is used to search for their synonyms and antonyms to expand the lexicon. The new words will be added into the seed list and the iteration will continue until no new word is found. However, this approach has a limitation, i.e. it cannot search for opinion words in a domain-specific orientation. Finally, in a corpus-based approach, a seed list is expanded via the help of a corpus text. Thus, it can help to search for domain-specific and oriented opinion words.

The works on sentiment analysis in business/finance news focus on using domain knowledge in the analysis. Godbole et al. (2007) investigated sentiment lexicon construction and analysis on news and blog [7]. The sentiment lexicon was initialized

through a seed list of polarity words, followed by the expansion of the seed list through synonym and antonym using WordNet. On the other hand, Ruiz-Martínez et al. (2012) proposed financial sentiment annotation using ontological resources with natural language processing resources [8]. The financial ontology was manually created for stock market domain. The open-source software GATE was used to annotate the sentiment using sentiment gazetteers developed to mark up all sentiment words and associated entities in our ontology. Another approach based on financial ontology was proposed by Salas-Zárate et al. [9]. Different from the previous approach, the polarity of each feature was identified based on the position of it within the text, words around the feature, and SentiWordNet. The sentiment polarity of the document was calculated by summing up all the scores from the features.

## 3 Methodology

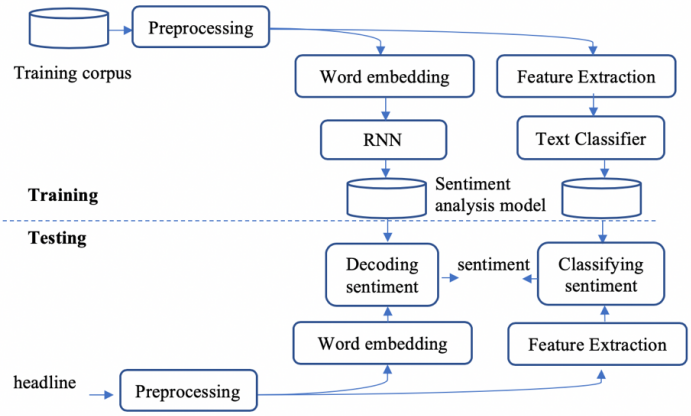


**Fig. 1.** Sentiment analysis steps

We examined two different types of approach for sentiment analysis: text classification approaches and recurrent neural networks (RNN). Both approaches went through similar preprocessing steps, but the text classification approaches required the text to be converted to a feature vector, while RNN required input in the form of word embedding vector before subsequent training/testing could be carried out. Fig. 1 shows the general process to perform sentiment analysis through machine learning.

Pre-processing was performed so that the relevant features could be extracted from the given texts. The preprocessing steps involved tokenization, part-of-speech (POS) tagging, lemmatization, stop words removal, and normalization. The tokenizer split each headline sentence into words using white spaces and punctuations as delimiters [10]. After tokenization, POS tagging of the tokens in each headline was optionally carried out for later use in lemmatization, which is an optional step that converts each token into its base word. This step helped to reduce the number of features by grouping similar words into the same base word form and ensured that the classifier generalizes better to other word forms which were not found in the training data but appeared in

the test data. Stop words were removed from the list of tokens, and normalization was performed by converting each token into lowercase, and numbers were converted to a tag using regular expressions.

| Feature | No Lemmatisation | | | | | With Lemmatisation | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Boolean Feature | a-g | ⋯ | malls | ⋯ | zwipe | a-g | ⋯ | mall | ⋯ | zwipe |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 1 | ⋯ | 0 | 0 | ⋯ | 1 | ⋯ | 0 |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 0 | ⋯ | 0 | 0 | ⋯ | 0 | ⋯ | 0 |
| Frequency | a-g | ⋯ | malls | ⋯ | zwipe | a-g | ⋯ | mall | ⋯ | zwipe |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 2 | ⋯ | 0 | 0 | ⋯ | 2 | ⋯ | 0 |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 0 | ⋯ | 0 | 0 | ⋯ | 0 | ⋯ | 0 |
| Probability | a-g | ⋯ | malls | ⋯ | zwipe | a-g | ⋯ | mall | ⋯ | zwipe |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 0.154 | ⋯ | 0 | 0 | ⋯ | 0.154 | ⋯ | 0 |
| | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ | ⋮ | ⋱ | ⋮ | ⋱ | ⋮ |
| | 0 | ⋯ | 0 | ⋯ | 0 | 0 | ⋯ | 0 | ⋯ | 0 |

**Fig. 2.** Snippets of feature matrices generated by various combinations of pre-processor and feature vector

In the feature extraction step, the bag-of-words model was used to represent the news headlines by treating each of them as a collection of words or tokens. Before such feature vector could be built, the vocabulary used, which is the set of unique words or tokens which appear in the headlines, must be learned from the training data first. The size of the vocabulary determines the dimension of the feature vector used to represent each headline. Each value in the feature vector corresponds to a unique word in the vocabulary. The value can be a Boolean feature (indicating whether the word appears in the headline sentence), a frequency value (indicating how many times the word appears in the headline sentence) or a probability value (indicating the probability of occurrence of the word in headline sentence, i.e. the frequency of the word divided by the total number of words in the headline sentence). Fig 2 shows snippets of the feature matrices for text classification. Four types of text classification model were experimented, namely multilayer perceptron (MLP) classifier, naïve Bayes classifier, decision tree classifier and support vector machine (SVM) classifier. Two types of naïve Bayes classifier, namely multinomial naïve Bayes classifier and complement naïve Bayes classifier were used as they are known to be suitable models for text classification [11]. These models were trained by passing in the feature vectors of the training data as well as the corresponding target vector of sentiment values.

On the other hand, for RNN, each token in a sentence was converted to a word embedding vector instead of a feature vector. First, each word is converted into an integer index which represents its frequency in the training dataset. Then, the Embedding layer in RNN will compute its vector representation.

## 4 Experiment and discussion

We are interested in predicting the sentiment of readers on business news headlines. To construct the sentiment corpus, we collected the data from the Internet. We used a web crawler to collect news headlines from *The Edge Markets*, one of the leading business news websites in Malaysia that many investors and traders read. We collected 60,000 headlines from the year 2014 until 2018. We applied a crowdsourcing approach to annotate the headlines. Each annotator would annotate the headline as either positive, negative or neutral. A headline is annotated as positive if it describes a positive sentiment about a company, security or market, and vice versa. On the other hand, a headline is annotated as neutral if it is neither positive nor negative. We managed to annotate 22,000 headlines: 20,000 headlines for training, 1,000 headlines for development and another 1,000 headlines for testing. Table 2 shows the distribution of the data.

**Table 2.** Distribution of different sentiments in the training and testing data

| Sentiment | Training Data | | Development Data | | Testing Data | |
|---|---|---|---|---|---|---|
| | Count | Percentage | Count | Percentage | Count | Percentage |
| Positive | 8,901 | 44.5% | 546 | 54.6% | 439 | 43.9% |
| Neutral | 6,838 | 34.2% | 217 | 21.7% | 305 | 30.5% |
| Negative | 4,261 | 21.3% | 237 | 23.7% | 256 | 25.6% |

Several libraries were used, including Natural Language Toolkit (NLTK) [10] for NLP tasks such as tokenization, POS tagging as well as lemmatization and Scikit-Learn [11] for text classification approaches. Table 3 shows the results from development set for each classifier based on the pre-processing and feature extraction steps used. Among them, multinomial naïve Bayes classifier yields the best accuracy of 66.6% when lemmatization is not applied and Boolean feature is used. In terms of accuracy, complement naïve Bayes classifier is as good as multinomial naïve Bayes classifier, followed by SVM classifier, while MLP classifier and decision tree classifier perform badly. It is worth noting that MLP classifier and decision tree classifier may generate different results when they are rerun as they incorporate some elements of randomness in their implementation, although their results are unlikely to differ much from time to time.

In general, the accuracy of the classifiers improves when lemmatization is applied. This shows that lemmatization is a useful step in ensuring that the trained model generalizes better to unseen data. In terms of features, Boolean and frequency features yield similar accuracies regardless of the type of classifier. This is probably due to their similar values as most tokens occur only once in a headline sentence. However, probability feature yields distinct results when they are used with different types of classifier. For instance, it yields the worst accuracy of 55.5% when it is used with MLP classifier. Also, multinomial naïve Bayes and SVM classifiers yield significantly lower accuracies when it is used, suggesting that it may not be suitable to be used with these classifiers. We selected the best model based on our development data result, the

multinomial naïve Bayes classifier to evaluate the testing data. The classifier yields an accuracy of **63.8%**.

**Table 3.** Accuracy (development data) of text classification approaches using different types of features in sentiment analysis

| Pre-Processing | No Lemmatization | | | With Lemmatization | | |
|---|---|---|---|---|---|---|
| Feature | Bool. | Freq. | Prob. | Bool. | Freq. | Prob. |
| MLP Classifier | 58.2% | 57.1% | **55.5%** | 60.7% | 57.8% | 57.9% |
| Multinomial Naïve Bayes | **66.6%** | 65.4% | 60.1% | 65.6% | 65.2% | 60.8% |
| Complement Naïve Bayes | 65.9% | 65.8% | 65.7% | 66.4% | 65.6% | 66.4% |
| Decision Tree | 57.2% | 58.1% | 57.1% | 59.8% | 58.2% | 57.0% |
| SVM | 62.2% | 62.7% | 58.9% | 64.4% | 64.4% | 60.1% |

Several strategies were attempted to improve the accuracy of the classifiers. One of such strategies is feature selection, which reduces the number of features by retaining only a subset of features that have the best discriminatory ability. The scoring of the features is done through chi-squared ($\chi^2$) test which determines the independence between each individual feature (token) and the sentiment class. Features that are not related to the sentiment class are removed from the feature vectors to improve the efficiency of the classifiers and hopefully their accuracy since the retained features are now more relevant and have better discriminatory ability.

For implementation, a built-in feature selector class in Scikit-Learn was used to select the $k$ best features in terms of their $\chi^2$ scores from the initial feature vectors. Consequently, the dimensions of the feature vectors were reduced from $n \times 14{,}352$ (no lemmatization) and $n \times 12{,}287$ (with lemmatization), where $n$ is the number of headlines, to $n \times k$. Table 4 shows the results of applying $k$-best feature selection on the feature vectors before classification. Generally, the accuracies of multinomial and complement naïve Bayes classifiers gradually decrease as $k$ decreases. This may be because as the number of features decreases, it becomes increasingly difficult for the classifiers to extract features from the given headlines. Nevertheless, complement naïve Bayes classifier achieves the highest accuracy of 66.8% when $k = 6{,}000$, and its corresponding accuracy from test set is **65.4%**, while MLP and SVM classifiers achieve particularly high accuracy when $k$ is the lowest. This suggests that feature selection can still improve the accuracy of certain types of classifiers.

Another method of feature selection is to select only the tokens with document frequency above a certain threshold value. Document frequency refers to the number of documents (headline sentences) in which the token appears. This step helps to remove rare words that usually do not carry any sentiment information, and thus prevents the model from overfitting. Table 5 shows the results of applying minimum document frequency (min-DF) filter to the feature vectors before classification, which reduces the size of the feature vector to 8,200 (no lemmatization) and 6,983 (with lemmatization) respectively. Generally, applying minimum document frequency filter reduces the accuracy of all the classifiers, except MLP classifier whose accuracy increases slightly.

Nevertheless, complement naïve Bayes classifier achieves the highest accuracy of 65.7% among all the models where minimum document frequency filter is applied. When evaluated using the testing data, this model yields an accuracy of **65.3%**.

**Table 4.** Accuracies (development data) of the classifiers using *k*-best feature selection

| Classifier | *k* | No Lemmatization | | | With Lemmatization | | |
|---|---|---|---|---|---|---|---|
| | | Bool. | Freq. | Prob. | Bool. | Freq. | Prob. |
| MLP Classifier | All | 58.2% | 57.1% | 55.5% | 60.7% | 57.8% | 57.9% |
| | 10,000 | 59.7% | 57.9% | 59.0% | 60.8% | 60.9% | 59.6% |
| | 6,000 | 55.8% | 57.1% | 57.7% | 60.2% | 59.0% | 60.3% |
| | 2,000 | 57.7% | 60.7% | 60.2% | 58.5% | 61.9% | 59.1% |
| Multinomial Naïve Bayes | All | 66.6% | 65.4% | 60.1% | 65.6% | 65.2% | 60.8% |
| | 10,000 | 66.5% | 66.1% | 59.9% | 65.7% | 65.4% | 60.3% |
| | 6,000 | 63.9% | 63.9% | 60.1% | 64.7% | 64.4% | 61.1% |
| | 2,000 | 63.0% | 63.2% | 60.1% | 64.5% | 64.1% | 60.8% |
| Complement Naïve Bayes | All | 65.9% | 65.8% | 65.7% | 66.4% | 65.6% | 66.4% |
| | 10,000 | 65.8% | 64.9% | 65.0% | 66.6% | 66.4% | 65.8% |
| | 6,000 | 63.6% | 63.0% | 64.3% | 65.4% | 65.3% | **66.8%** |
| | 2,000 | 63.1% | 63.7% | 62.5% | 64.3% | 64.0% | 64.4% |
| Decision Tree | All | 57.2% | 58.1% | 57.1% | 59.8% | 58.2% | 57.0% |
| | 10,000 | 57.7% | 57.1% | 55.9% | 55.8% | 58.0% | 59.5% |
| | 6,000 | 58.0% | 57.2% | 55.9% | 57.6% | 58.1% | 57.8% |
| | 2,000 | 56.9% | 56.8% | 55.4% | 57.0% | 56.2% | 55.5% |
| SVM | All | 62.2% | 62.7% | 58.9% | 64.4% | 64.4% | 60.1% |
| | 10,000 | 63.1% | 63.5% | 58.4% | 64.9% | 64.6% | 59.9% |
| | 6,000 | 60.2% | 60.8% | 58.2% | 64.4% | 64.2% | 59.2% |
| | 2,000 | 64.0% | 64.2% | 57.7% | 65.0% | 66.0% | 59.5% |

**Table 5.** Accuracies (development data) of the classifiers using min-DF filter of 2

| Classifier | Min-DF | No Lemmatization | | | With Lemmatization | | |
|---|---|---|---|---|---|---|---|
| | | Bool. | Freq. | Prob. | Bool. | Freq. | Prob. |
| MLP Classifier | None | 58.2% | 57.1% | 55.5% | 60.7% | 57.8% | 57.9% |
| | 2 | 58.5% | 58.1% | 57.9% | 60.0% | 59.1% | 58.3% |
| Multinomial Naïve Bayes | None | **66.6%** | 65.4% | 60.1% | 65.6% | 65.2% | 60.8% |
| | 2 | 64.7% | 64.5% | 61.2% | 64.5% | 64.2% | 61.7% |
| Complement Naïve Bayes | None | 65.9% | 65.8% | 65.7% | 66.4% | 65.6% | 66.4% |
| | 2 | 64.8% | 64.5% | 63.9% | 65.1% | 64.3% | **65.7%** |
| Decision Tree | None | 57.2% | 58.1% | 57.1% | 59.8% | 58.2% | 57.0% |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 2 | 55.9% | 58.8% | 54.1% | 59.0% | 57.6% | 57.6% |
| SVM | None | 62.2% | 62.7% | 58.9% | 64.4% | 64.4% | 60.1% |
| | 2 | 62.3% | 61.7% | 58.7% | 64.4% | 64.1% | 60.4% |

For testing the recurrent neural networks approach, we used the same data for training, testing and development. We examined two types of RNN architecture. One using a typical RNN architecture, and another applied the encoder-decoder architecture [12]. In both the architectures, a specialized RNN cell, which is the long-short term memory (LSTM). The typical RNN architecture consists of 4 hidden layers of LSTM cell. The first, second, third and fourth layer consists of 128, 64, 32, and 16 states correspondingly. The Softmax activation method is used in output layer since this is a multiclass classification. We also tested the LSTM encoder-decoder architecture that is normally used for sequence-to-sequence modelling such as neural machine translation [13]. The encoder uses a bidirectional cells with attention. Both the encoder and decoder consists of one hidden layer, and 128 states. The output layer of the decoder is also using the Softmax activation method. The result is presented in Table 6. The result shows that the encoder-decoder architecture is slightly better than the typical RNN architecture in sentiment analysis. If we compare both the results of text classification approach and the RNN approach, we can see that the encoder-decoder approach is slightly better.

**Table 6.** Accuracies (test data) of RNN approaches

| RNN | No Lemmatization | With Lemmatization |
|---|---|---|
| LSTM (4 layers) | 63.9% | 64.1% |
| Bidirectional LSTM encoder-decoder with attention | **66.9%** | **-** |

## 5 Conclusion

In this work, we collected and annotated a sentiment analysis corpus that consists of business news headlines. We evaluated two different approaches, namely the text classification approach and recurrent neural networks in modelling and predicting the sentiments of headlines. In the text classification approach, we tested the MLP classifier, multinomial naïve Bayes, complement naïve Bayes and decision trees approaches. The complement naïve Bayes approach with k-best feature selection gave the highest accuracy at 65.4% on the test data using the feature vector that consists of lemma and probabilities. The recurrent neural network approach using bidirectional encoder-decoder architecture with attention on the other hand obtained an accuracy of 66.9%. Thus, in term of accuracy, the neural network approach has a slight lead. The second advantage of RNN approach is it does not need any intervention in feature selection. Thirdly, the syntactic information has been removed in the bag of word approach, but not in the RNN approach. Thus, we foresee if more annotated data is available, the RNN approach will perform even better. However, in term of time used for modeling

and testing, the RNN approach took a longer time than the text classification approach even using GPU.

## Acknowledgement

This work is funded by Universiti Sains Malaysia through the Bridging grant scheme 304.PKOMP.6316283.

## References

1. Kušen, E., Strembeck, M.: Politics, sentiments, and misinformation: An analysis of the Twitter discussion on the 2016 Austrian Presidential Elections. Online Social Networks and Media 5: 37-50 (2018).
2. Miranda, M. D., José Sassi, R.: Using sentiment analysis to assess customer satisfaction in an online job search company. Lecture Notes in Business Information Processing 183: 17-27 Springer Cham (2014).
3. Kaushik A, Kaushik A, Naithani S.: A Study on Sentiment Analysis: Methods and Tools. International Journal of Science and Research (IJSR) *4*, 287-292 (2015).
4. Kiritchenko S., Zhu, X., Cherry, C., Mohammad, S. M.: Detecting aspects and sentiment in customer reviews. In: 8th International Workshop on Semantic Evaluation (SemEval), pp. 437-442, ACL, Dublin (2014).
5. M. Alharbi, A. M., de Doncker, E.: Twitter sentiment analysis with a deep neural network: An enhanced approach using user behavioural information. Cognitive Systems Research 54, pp. 50-61 (2019).
6. Zhang, L., Wang, S., Liu, B.: Deep learning for sentiment analysis: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery (2018).
7. Godbole, N., Srinivasaiah M., and Skiena, S.: Large-scale sentiment analysis for news and blogs. In: ICWSM'2007, Boulder (2007).
8. Ruiz-Martínez, J. M., Valencia-García, R., García-Sánchez, F.: Semantic-based sentiment analysis in financial news. In: 1st International Workshop on Finance and Economics on the Semantic Web, pp. 38–51, Heraklion (2012).
9. Salas-Zárate, M. P., Valencia-García, R. Ruiz-Martínez, A., Colomo-Palacios, R.: Feature-based opinion mining in financial news: An ontology-driven approach. Journal of Information Science 43(4): 458-479 (2017).
10. Bird, S., Loper, E., Klein E.: Natural Language Processing with Python. O'Reilly Media Inc., Sebastopol (2009).
11. Pedregosa F., Gaël, V., Alexandre, G., Vincent, M., Bertrand, T., Olivier, G., Mathieu, B., Peter, P., Ron, W., Vincent, D., Jake, J., Alexandre, P., David, C., Matthieu, B., Matthieu, P., and Édouard, D..:Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research: 12: 2825-2830 (2011).
12. Géron, A.: Hands-on machine learning with scikit-learn and tensorflow: concepts, tools, and techniques to build intelligent systems, O'Reilly Media, California (2017).
13. Bérard, Alexandre, Olivier Pietquin, Christophe Servan, and Laurent Besacier.: Listen and translate: A proof of concept for end-to-end speech-to-text translation. In: NIPS: pp. 1–5, Barcelona (2016).