SENTI2VEC: AN EFFECTIVE FEATURE EXTRACTION TECHNIQUE FOR SENTIMENT ANALYSIS BASED ON WORD2VEC

Eissa M.Alshari[1], Azreen Azman[2*], Shyamala Doraisamy[3], Norwati Mustapha[4] and Mostafa Alksher[5]

[1] Ibb University, Yemen

[2,3,4,5] Universiti Putra Malaysia, Serdang, Malaysia

Email: eissa.alshari@student.upm.edu.my[1], azreenazman@upm.edu.my[2] (corresponding author), shyamala@upm.edu.my[3], norwati@upm.edu.my[4], mostafa.alksher@student.upm.edu.my[5],

DOI: https://doi.org/10.22452/mjcs.vol33no3.5

ABSTRACT

The discovery of an active feature extraction technique has been the focus of many researchers to improve the performance of classification methods, such as for sentiment analysis. Many of them have shown interest in using word embeddings especially Word2Vec as the features for text classification tasks. Its ability to model high-quality distributional semantics among words has contributed to its success in many of the functions. Despite the success, Word2Vec features are high dimensional that lead to an increase in the complexity of the classifier. In this paper, an effective method for feature extraction based on Word2Vec is proposed for sentiment analysis. The process discovers polarity clusters of the terms in the vocabulary through Word2Vec and opinion lexical dictionary. The features vector for each text is constructed from the polarity clusters, which lead to a lower-dimensional vector to represent the text. This paper also investigates the effect of two opinion lexical dictionaries on the performance of sentiment analysis, and one of the dictionaries are created based on SentiWordNet. The effectiveness of the proposed method is evaluated on the IMDB with two classifiers, namely the Logistic Regression and the Support Vector Machine. The result is promising, showing that the proposed method can be more effective than the baseline approaches.

Keywords: Sentiment analysis, SentiWordNet, Word2Vec, Word embeddings

1.0 INTRODUCTION

Due to the increasing growth in online shopping, many Internet users find it very difficult to decide on their shopping needs. The users have to evaluate many similar products with different features, quality, and prices before making a purchase decision. Unlike offline shopping, where the customers can determine the products physically, the users are often relying on promotional images or videos to make a decision.

More recently, online shopping websites, such as Amazon.com, allow Internet users to give a rating for products that are being sold on the site. The score will indicate the degree of satisfaction of the users for particular products. Those websites also provide facilities for users to post comments regarding the products [1]. Such comments can represent their opinion on different aspects of products. As such, the ratings and comments can be used by Internet users as an additional recommendation to help them in making purchasing decisions [2, 3]. Due to the vast number of different opinions on a specific product, a user may find it difficult to summarize the overall sentiment based on those reviews or comments.

The problem of sentiment analysis is to determine the polarity of text to either positive, negative, or neutral [4]. Over the years, researchers have developed different techniques for sentiment analysis to classify the reviews or comments into their polarity classes [5, 6]. Many machine-learning methods for classification such as Naive Bayes (NB), Support Vector Machine (SVM) and Logistic Regression (LR) have shown to be useful in this text classification problem [7].

Those techniques rely on the features used to represent the text in the classification task. Thus, the effectiveness of any classification-based sentiment analysis approach depends on the ability of the features vector to discriminate among the polarity classes. Several features have been investigated for this task such as the bag-of-words (BoW), lexical, and syntactic features [8, 9], and word embeddings [10]. In [11, 12], the authors introduce the Word2Vec model to discover the semantic relation between words, and it is useful as features for several text classification

tasks. Due to this breakthrough, many efforts have been spent to apply Word2Vec for sentiment analysis. However, the Word2Vec features are typically high dimensional. Thus it can increase the complexity of any classifiers. As such, several feature extraction methods have been applied to reduce the dimension of the Word2Vec features [13].

This paper proposes a method to construct a feature set to reduce the dimension of the Word2Vec features for sentiment analysis. In many techniques for sentiment analysis, opinion lexical dictionary is often used as additional resources in constructing the features for classification. The list of positive and negative terms in the glossary helps the classifier to discriminate a text for positive or negative polarity. In the proposed method, the set of terms in vocabulary are clustered around opinion words from the opinion lexical dictionary to distribute the terms based on polarity. As such, all terms in the vocabulary will have its polarity alignment based on their semantic relation with other opinion words. Then, a features vector is constructed for each text based on the polarity clusters.

This paper is organized as follows. A review of related work on sentiment analysis and word embeddings is presented in Section 2. The proposed feature extraction method based on Word2Vec, the Senti2Vec, is explained in Section 3. In Section 4., the experimental results are analyzed and elaborated. Finally, the conclusion and future work are discussed in Section 5.

2.0 RELATED WORK

Sentiment analysis (SA) is a collection of methods to determine the polarity or orientation (positive, negative, or neutral) of a sequence of words in the text [14]. Many techniques and types of features have been investigated for SA including the use of bag-of-words (BoW) model as the feature for the classification [15]. The BoW is an approach to model texts numerically in many text mining and information retrieval tasks [16]. Several weighting schemes have been successfully used in the BoW such as the *n*-gram, Boolean, co-occurrence, *tf,* and *tf .idf* [6, 17].

Many researchers have focused their interest in modeling distributional semantics within the text as a weighting scheme. Several models based on the discrete representation of words have been proposed such as the Latent Semantic Analysis (LSA) [18], the Latent Dirichlet Allocation (LDA) [19], the Second Order Attributes (SOA) [20] and the Document Occurrence Representation (DOR) [21]. Villegas *et al*. [23] compared the word embedding approach for sentiment analysis by using several weighting schemes including *tf .idf* and Boolean on a subset of the IMDB Review Dataset. They found out that the effectiveness of the LSA as features with Naive Bayes classifier outperforms other techniques [23].

In [24], Giatsoglou *et al*. observed that LDA is computationally very expensive as compared to LSA on large datasets. In contrast, Fan *et al*. used Naive Bayes as the classification method to build sentiment lexicons through word vectors matrices separately and then used the Boolean rules to classify the matched documents for polarity that appeared in both matrices [38]. An extended model for sentiment classification [16], is presented by Haocheng *et al*. in [42], which focused on the semantic features between words rather than the simple lexical or syntactic features. For micro-blog, Zhang *et al*. investigated the use of multi-label classification, two micro-blog datasets, and eight different evaluation matrices on three different sentiment dictionaries [43]. In [44], the document vector was utilized to generate a labeled dataset by using an unsupervised learning approach based on labeled training datasets. Furthermore, an analysis with different classifiers (SVM, Naive Bayes, and Maximum Entropy) showed that the unigram with SVM performed the best [40]

Recently, there is also an increasing interest in modeling continuous representation of words based on artificial neural networks (ANN) such as the Word2Vec [11] and the GloVe [22]. Its application on sentiment analysis has been encouraging in which a significant difference in the performance can be observed for Word2Vec with SVM^{perf} classifier based on a dataset of Chinese comments for clothing products [41]. Historically, the ANN was introduced in 1943 [25], and the learning of continuous vector representation began with the introduction of the training for the feed-forward network in 1986 [26]. The authors in [26] modeled multiplying weights and added biases to initial inputs with the gradients to train the vectors. Since then, a common framework for estimating neural network language model (NNLM) was proposed in [27], where a feed-forward neural network with a linear projection layer and a non-linear hidden layer were used to learn the word vector representation and statistical language model. On the other hand, another architecture for learning word vectors by using the neural network with a single hidden layer was proposed such that the word vectors are digested even without constructing the full NNLM [28]. Consequently, the word vectors can be applied to significantly improve and simplify many natural language processing (NLP) applications [29]. Furthermore, the estimation of the word vectors has been done by using different model architectures and trained on various corpora [29, 30]. However, based on our knowledge, these architectures were significantly expensive for training.

Deep learning algorithms have been used to learn the representation of sentences but have been more successful in discovering syntactic rules of longer text as compared to shorter text [35, 36, 37]. Researchers have actively investigated its application in sentiment analysis based on several neural network models such as (Recursive Neural Network (RNN)[33], Convolutional Neural Network (CNN[27], and Long Short-Term Memory (LSTM)[34]). Mikolov *et al*. (2013) proposed Word2Vec model with an argument that a high-quality representation can be trained from huge data sets with billions of words in the vocabulary. They developed a new model that preserved the linear regularities around terms and achieved high accuracy for vector operations. In fact, the Word2Vec is not exactly a deep learning model to estimate the polarity of sentiment analysis. However, several studies have been using Word2Vec to normalize the inputs vectors and extracting further features. Mikolov *et al*. (2014) found that artificial neural network performed better than LSA for preserving linear regularities around words.

Furthermore, a combination of sentiment lexicon and Word2Vec is investigated to add more features to the classification matrix to extract extra syntax and semantics features from words. Le *et al*. proposed a deep learning model called a Paragraph Vector approach (Doc2Vec) for representing vectors as the length of texts such as paragraphs, sentences, and documents [16]. To evaluate the effectiveness of Doc2Vec, Lau *et al*. used the Word2Vec with an *n*-gram model to construct both the Distributed Bag of Words version of Paragraph Vector (DBoW) and the Distributed Memory version of Paragraph Vector (DMPV) for the Doc2Vec [39]. The results showed that DBoW is better than the DPMV model.

3.0 SENTI2VEC: FEATURE EXTRACTION METHOD BASED ON POLARITY CLUSTERS FOR WORD2VEC

The performance of a classification-based approach for sentiment analysis depends on the features used to represent the text for the classification algorithm such as Support Vector Machine (SVM) or Logistic Regression (LR). The models such as the Bag of Words (BoW) or the Word2Vec [33, 45] have shown to be useful in this task. However, the BoW model typically does not take into consideration the semantic relationships among words and the features created from Word2Vec model are often high-dimensional that leads to inefficient classification.

This paper proposes a feature extraction method based on modeling polarity clusters within the Word2Vec vectors to improve the effectiveness of sentiment analysis. It is assumed that each word in the vocabulary has its polarity alignment and will produce a better representation of text for sentiment analysis. The method proposed in this paper consists of three main components. (1) the learning of word embeddings based on Word2Vec, (2) the discovery of polarity clusters based on opinion lexical dictionary, and (3) the construction of features matrix for classification based on cluster centroids as shown in Fig 1.

3.1. Building opinion lexical dictionary from sentiWordNet (SWN)

The method described in this paper relies on an opinion lexical dictionary to construct clusters of terms in the vocabulary. There are several opinion lexical dictionary available to support many sentiment analysis techniques [46]. An opinion lexical dictionary consists of a list of positive terms (e.g., *like* and *love*) and negative terms (e.g., *bad* and *disappointed*). In this paper, an opinion lexical dictionary consists of 6,788 terms (4,783 negative terms (70.5%), and 2,005 positive terms (29.5%) are used, and the dictionary has successfully used in many sentiment analysis problems [47].

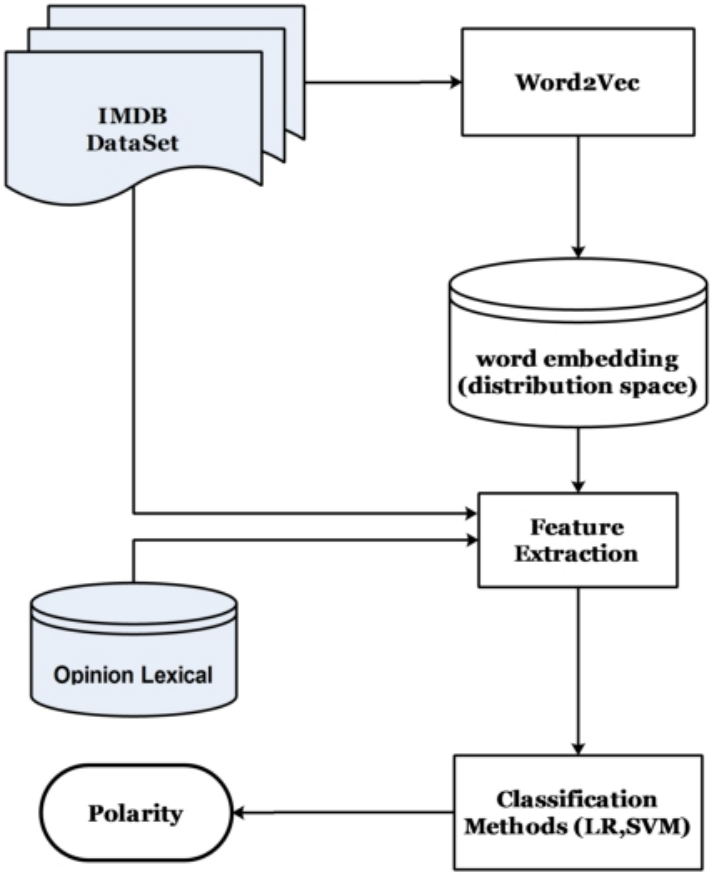

Fig. 1. Framework for the proposed method

Another source of opinion lexical dictionary often used in sentiment analysis is the SWN ([48, 49]). The size of SWN is larger and has a total of 191,000 terms. It is also more flexible as a term can be both positive and negative with assigned polarity scores. For instance, *like* has a score of 0.9 for positive and 0.1 for negative. To use the SWN for this proposed method, the polarity of each term is determined based on which polarity score is the largest, such that *like* will be positive because 0.9 is higher than 0.1. As a result, a total of 38,063 terms are identified as opinion words with 18,121 positive terms (47.6%) and 19,942 negative terms (52.4%), creating a more balanced opinion lexical dictionary.

Data: IMDB with 100K documents
Result: Vector representation of terms
begin
  foreach $doc \in corpus$ **do**
    foreach $term \in doc$ **do**
      $terms \leftarrow$pre-processing all $term$ with NLP methods;
    end
    $vocabulary \leftarrow terms$
  end
  foreach $term \in vocabulary$ **do**
    $vector_{term} \leftarrow w2vec_{term}$
  end
  foreach $word_{neg}, word_{pos} \in Dict_{neg}, Dict_{pos}$ **do**
    $word_{neg}v \leftarrow vector_{w2vec}$
    $word_{pos}v \leftarrow vector_{w2vec}$
  end
  foreach $term \in vocabulary$ **do**
    $value_p \leftarrow max(sim(term, word_{pos}))$
    $value_n \leftarrow max(sim(term, word_{neg}))$
    if $value_p > value_n$ **then**
      $vector_{term} \leftarrow value_p$
      $word_{pos} \leftarrow term, vector_{term}$
      else
        $vector_{term} \leftarrow value_n$
        $word_{neg} \leftarrow term, vector_{term}$
      end
    end
    $newvector_{term} \leftarrow vector_{term}$
  end
end

Algorithm1: The algorithm for the proposed method

3.2. Learning Word Representation based on Word2Vec

As shown in Algorithm 1, the first component of the method deals with the discovery of word representation based on the Word2Vec model. Given a corpus D consists of a set of texts, D= {$d1$, $d2$, $d3$,..., dn}, and a vocabulary T = {$w1$, $w2$, $w3$,..., wm}, which consists of unique terms extracted from D. The word representation of the terms w_i is discovered by using the Skip-gram model of the Word2Vec [16] to calculate the probability distribution of other terms in the context given w_i. In particular, w_i is represented by a vector $\overrightarrow{v_i}$ that comprises of probabilistic values of all terms in the vocabulary. This word embedding technique can discover semantic relationships among terms in the corpus. However, the resulting set of vectors for all terms in the corpus is high-dimensional and is inefficient for the classifier in the sentiment analysis task. As a result, this first component discovers a set of vectors $V_T = \{\overrightarrow{v_1}, \overrightarrow{v_2}, \overrightarrow{v_3}, ..., \overrightarrow{v_m}\}$ representing the set of terms in the vocabulary T.

$$T = \begin{bmatrix} w_1 \\ w_2 \\ . \\ . \\ w_j \end{bmatrix} \Longrightarrow \vec{V}_T = \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ . \\ . \\ \vec{v}_i \end{bmatrix} \quad (1)$$

3.3. Discovery of polarity clusters within Word2Vec based on opinion lexical dictionary

Constructing a feature matrix directly from the word representation produced by the Word2Vec model is inefficient as it tends to be massive due to the high-dimensional nature of the word representation. In this paper, the terms t_i in the vocabulary is grouped into a set of clusters which used to represent the text as a feature vector. As this paper focuses on sentiment analysis, it is rational to assume that the groups should be generated based on the polarity of the terms in the vocabulary. However, most of the terms in the vocabulary are non-opinion words and do not have sentiment polarity. Therefore, all terms in T are clustered based on their similarity to the set of opinion words. As such, the polarity of non-opinion words is estimated based on the cluster they are in.

To cluster the terms based on polarity, an opinion lexical dictionary that contains a list of opinion words is chosen as the centroid of the clusters. The aim is to group non-opinion words in the vocabulary into several clusters of opinion words obtained from the opinion lexical dictionary. Let S be a set of opinion words (both negative and positive) from the opinion lexical dictionary, and C be the set of centroid terms, which consists of those common terms in the dictionary that also appears in the vocabulary, $C = T \cap S$. Due to the curse of dimensionality problem of language model training, some of the terms in the opinion lexical dictionary do not appear in the vocabulary of the Word2Vec. Thus, these words are ignored and are not used as the centroid.

In this paper, two sets of opinion lexical dictionary are used as described in Section 3.1. The first dictionary consists of 2,005 of positive terms and 4,783 of negative terms and has been proven to be useful in many sentiment analysis techniques [50]. Since the number of terms in the dictionary is small, almost 600 terms are ignored from being chosen as the centroids. The second dictionary is extracted from the SWN as described in Section 3.1. In this case, there are 18,121 positive terms, and 19,942 negative terms are chosen as the centroids creating more clusters as compared to the first dictionary. It is assumed that a more significant number of clusters will produce a better representation of text as the features for the classification task. For this dictionary, around 6,000 terms are not chosen as the centroids.

As each term in both the vocabulary $w_i \in T$ and the centroid $w_j \in C$ are represented by vectors as described in Section 3.2., the similarity between both terms can be estimated by using the *cosine* similarity of their respective vector representation such that, $sim(w_i, w_j) = cosine(\overrightarrow{v_i}, \overrightarrow{v_i})$. Therefore, for each term in T, its similarity with all terms in C are calculated, and the term is assigned to a cluster C_j if $sim(w_i, w_j)$ is the maximum for the term w_i and w_j is the centroid of the cluster C_j. As a result, all terms in the vocabulary are clustered based on the opinion words in the dictionary. Each cluster C_j will consist of a set of tuples representing the members of the cluster, such that each tuple consists of the term and its similarity to the centroid of the cluster, w_i, $sim(w_i, w_j)$.

$$C_j = \{\langle w_1, sim(w_1, w_j)\rangle, \langle w_2, sim(w_2, w_j)\rangle, ..., \langle w_i, sim(w_i, w_j)\rangle\} \quad (2)$$

Note that, since all the opinion words in the vocabulary T are used as the centroids, they are automatically part of a cluster and no tuples added for them.

3.4. Feature extraction based on polarity clusters

The third component of the model aims to construct a feature vector for each text $d_i \in D$ based on the clusters discovered in the previous step. Therefore, instead of using the entire vocabulary size as the dimension of the feature vector, this approach will limit the dimension of the vector to the number of the cluster, $|C|$. To construct the feature vector vd_i for a given text d_i, all the terms in d_i are scanned, and the clusters that those terms belonged into are selected as the features to represent d_i. The weight is assigned to the column representing the cluster and is given by the similarity score between the centroid of the cluster, and its members. As there could be more than one member in the cluster, only a single value of similarity score is selected for the weight. In this paper, the maximum similarity score of the given cluster is chosen as the weight for the feature. It has shown to be more effective than other scores, such as minimum or average.

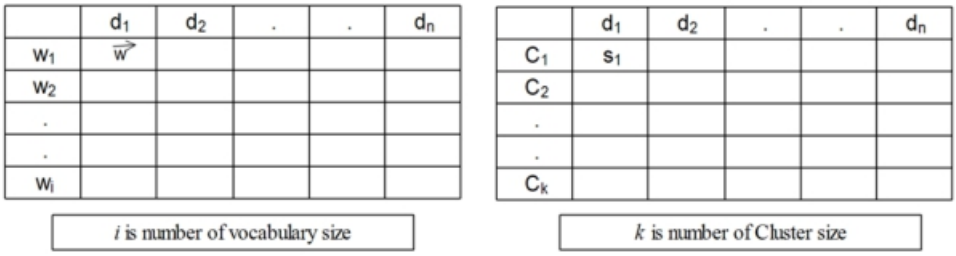

Fig. 2. Size comparison of the feature matrix

Then, a feature matrix is constructed by combining all feature vectors for D into a matrix. As such, the resulting feature matrix of size $n \times |C|$ will be smaller than a typical feature matrix constructed based on bag-of-words, which is normally $n \times m$, due the dimension is limited to the number of clusters generated based on the proposed method. It is also expected that such reduction will have a positive impact on the overall effectiveness of the sentiment analysis.

4.0 EXPERIMENTAL RESULTS AND ANALYSIS

The effectiveness of the proposed model in the sentiment analysis task is evaluated by using the Large Movie Review Dataset (ACLIMDB), which is available on-line[1]. The dataset consists of 100,000 movie reviews, and 50,000 of the reviews are labeled [51]. The dataset has been used in many sentiment analysis evaluations. All experiments were performed on an Intel i7-7700 3.60GHz PC with NVIDIA GeForce GTX 1070 GPU and 16GB of RAM.

In this experiment, the features extracted based on this model will be used in two classification methods, namely the Logistic Regression (LR) and the Support Vector Machine (SVM) to compare the performance in the different classifiers. Also, two different opinion lexical dictionaries are used in this experiment as mention in Section 3.1. The first dictionary, *Dic*, is smaller with only 6,788 terms and imbalanced distribution of positive (29.5%) and negative (70.5%) terms [13, 47]. The second dictionary, *senti*, is constructed from the SWN based on the method described in Section 3.1. The dictionary is bigger with 38,063 terms and evenly distributed with 47.6% positive terms and 52.4% negative terms. This experiment attempts to discover the effect of the dictionary in the performance of the model for sentiment analysis.

As discussed in Section 3.4., the weights assigned to each feature vectors are only based on the similarity of each term in the text and the centroid of the cluster it belonged to. As such, the discriminative power of the term within the corpus will be ignored. As a comparison, this experiment attempts to discover the effect by multiplying the similarity value with the normalized *idf* of the term calculated based on the corpus.

The performance of the proposed method for sentiment analysis is compared based on the classification accuracy measure against the Word2Vec [12], the Doc2Vec [44] and the Bag-of-words [51, 40] methods. The Doc2Vec is an unsupervised algorithm to generate vectors for paragraphs, sentences, comments or documents that is an adaptation of Word2Vec which can create vectors for words. The description of different approaches being evaluated in this paper is depicted in Table 1.

1 http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz

Table 1: The description of the approaches

Abbreviation	Description
BoW_tfidf	Bag-of-words model with a *tf.idf* weighting scheme
Doc2Vec	Doc2Vec model, an improvement of Word2Vec [44]
Senti2Vec_dic	The proposed model with *dic* as a dictionary (Section 3.1.)
Senti2Vec_senti	The proposed model with *senti* as a dictionary (Section 3.1.)
Senti2Vec_dic + *idf*	The proposed model with *dic* as dictionary multiplied by normalized *idf* as the weight
Senti2Vec_senti + *idf*	The proposed model with *dic* as dictionary multiplied by normalized *idf* as the weight

The experiment is conducted by using 10-folds cross-validation technique to evaluate a predictive model by partitioning the data into ten equal size sets, and at each iteration, nine sets of data are used for training, and one set is used for testing. The experiment is repeated by selecting different sets of test data from the sets. As such, the results can be generalized to an independent data set. Table 2 shows the accuracy of varying feature models for sentiment analysis based on LR and SVM classifiers. The baseline for this comparison is the BoW tfidf and the Doc2Vec methods with the accuracy of 87.3% and 87.4% for LR classifiers, respectively. In the case of SVM, the models obtain an accuracy of 84.4% and 86.1%, respectively. It shows that using Word2Vec based features for sentiment analysis is comparable with the standard BoW model.

For the LR classifier, the proposed model with opinion dictionary, Senti2Vec_dic, performs better than the baseline, Doc2Vec, with an accuracy of 90.8%, which is an improvement of about 4%. When a larger opinion lexical dictionary based on SWN, Senti2Vec_senti, is used, the accuracy is further improved to 93.4%, an improvement of about 7%. As such, it shows that a more significant dictionary that creates more clusters to be used for the construction of the feature vectors will have a positive effect on the performance of the proposed model. However, the inclusion of *idf* to add the discriminative power of the vector does not improve the performance of the proposed model. The accuracy drops to 88.1% for the Senti2Vec_dic model and 78.7% for the Senti2Vec_senti.

Table 2: The accuracy of different approaches based on LR and SVM classifiers

Classifier	Approach	Precision	Recall	Accuracy (%)	Time (s)
LR	BoW_tfidf	0.875	0.875	87.3	2085
	Doc2Vec	0.870	0.870	87.4	1439
	Senti2Vec_dic	0.905	0.910	90.8	197
	Senti2Vec_senti	0.935	0.935	93.4	208
	Senti2Vec_dic + *idf*	0.875	0.875	88.1	555
	Senti2Vec_senti + *idf*	0.790	0.790	78.7	568
SVM	BoW_tfidf	0.845	0.845	84.4	2893
	Doc2Vec	0.860	0.860	86.1	1551
	Senti2Vec_dic	0.785	0.780	78.2	234
	Senti2Vec_senti	0.830	0.840	84	217
	Senti2Vec_dic + *idf*	0.795	0.795	79.5	612
	Senti2Vec_senti + *idf*	0.625	0.620	67.5	599

A different scenario is observed for the SVM classifier where in general the proposed model is not better than the baseline. For instance, the accuracy of the Senti2Vec_dic model is only 78.2% and 84.0% for the Senti2Vec_senti model. Nevertheless, the pattern is consistent whereby more clusters produced by a bigger size of opinion lexical dictionary will improve the performance of the sentiment analysis.

Also, it is observed that the proposed method decreases the size of the feature set to almost 80% of the Word2Vec size, which will reduce the complexity of the classifier. As observed based on Table 2, the proposed Senti2Vec features are more efficient as compared to the BoW and Doc2Vec features in which the processing of the best of the Senti2Vec (Senti2Vec_senti) is 90% faster than the BoW and 86% faster than the Doc2Vec based on LR classifier. A similar pattern is observed for the SVM classifier where 92% and 86% improvement is shown against the BoW and the Doc2Vec, respectively. As a result, the proposed method will be more effective as well as efficient for sentiment analysis.

5.0 CONCLUSION

This paper proposes a method to reduce the dimension of the features vector based on the Word2Vec for sentiment analysis. Clusters of terms centered by a set of opinion words from two opinion lexical dictionaries are constructed. A simple transformation is applied to the negative term vectors to redistribute the terms in the space based on their polarity. As such, a much smaller matrix of document vectors is produced based on the set of clusters. Two classifiers, namely Logistic Regression and Support Vector Machine (SVM) are used to compare the performance of different feature set for sentiment analysis.

It has been observed that the performance of the proposed method is encouraging, showing that it can be more effective and efficient than the baseline. In the future, more investigation will be performed on the Word2Vec in term of the perplexity. As the proposed method relies on an opinion lexical dictionary to construct the clusters, it will be more flexible if no dictionary is required to build the clusters.

ACKNOWLEDGEMENT

This work is supported by the Ministry of Higher Education Malaysia under the FRGS Grant (FRGS/1/2015/ICT04/UPM/02/5) and Universiti Putra Malaysia under the *Geran Putra* (GP/2017/9588800).

REFERENCES

[1] S. Shojaee and A. bin Azman, "An evaluation of factors affecting brand awareness in the context of social media in Malaysia," *Asian Social Science*, vol. 9, no. 17, p. 72, 2013.

[2] J. B. Schafer, J. A. Konstan, and J. Riedl, "E-commerce recommendation applications," in *Applications of Data Mining to Electronic Commerce*. Springer, 2001, pp. 115–153.

[3] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts *et al.*, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[4] Y. Chen, B. Perozzi, R. Al-Rfou, and S. Skiena, "The expressive power of word embeddings," *arXiv preprint arXiv:1301.3226*, vol. 28, pp. 2–9, 2013.

[5] A. Graves, S. Fern´andez, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *International Conference on Artificial Neural Networks*. Springer, 2005, pp. 799–804.

[6] X. Lui and W. B. Croft, "Statistical Language Modeling For Information Retrieval," *Annual Review of Information Science and Technology 2005 Volume 39*, vol. 39, p. 1, 2003.

[7] Z. Yu, H. Wang, X. Lin, and M. Wang, "Learning term embeddings for hypernymy identification." in *IJCAI*, 2015, pp. 1390–1397.

[8] X. Liu and W. B. Croft, "Statistical language modeling for information retrieval," *Annual Review of Information Science and Technology*, vol. 39, no. 1, pp. 1–31, 2005.

[9] S. Shojaee, M. A. A. Murad, A. B. Azman, N. M. Sharef, and S. Nadali, "Detecting deceptive reviews using lexical and syntactic features," in *Intelligent Systems Design and Applications (ISDA), 2013 13th International Conference on*. IEEE, 2013, pp. 53–58.

[10] F. Enr´ıquez, J. A. Troyano, and T. L´opez-Solaz, "An approach to the use of word embeddings in an opinion classification task," *Expert Systems with Applications*, vol. 66, pp. 1–6, 2016.

[11] 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*, 2013, pp. 3111–3119.

[12] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[13] M. Hu and B. Liu, "Mining and summarizing customer reviews," *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining KDD 04*, vol. 04, p. 168, 2004.

[14] J. Kim, J. Yoo, H. Lim, H. Qiu, Z. Kozareva, and A. Galstyan, "Sentiment Prediction using Collaborative Filtering," *Icwsm*, 2013.

[15] R. Nikhil, N. Tikoo, S. Kurle, H. S. Pisupati, and G. R. Prasad, "A survey on text mining and sentiment analysis for unstructured web data," in *Journal of Emerging Technologies and Innovative Research*, vol. 2, no. 4 (April-2015). JETIR, 2015, pp. 1292–1296.

[16] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," pp. II–1188, 2014.

[17] E. Emad, E. M. Alshari, and H. M. Abdulkader, "Arabic Vector Space Model based on Semantic," in *International journal of computer science (IJISI),*, vol. 8, no. 6. Ain Shams, 2013, pp. 94–101.

[18] T. K. Landauer and S. T. Dumais, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge." *Psychological Review*, vol. 104, no. 2, p. 211, 1997.

[19] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent Dirichlet allocation," in *advances in neural information processing systems*, 2010, pp. 856–864.

[20] A. P. L´opez-Monroy, M. Montes-Y-Gomez, H. J. Escalante, L. V. Pineda, and E. Villatoro-Tello, "Inaoe's participation at pan'13: Author profiling task notebook for pan at clef 2013." in *CLEF (Working Notes)*, 2013.

[21] A. Lavelli, F. Sebastiani, and R. Zanoli, "Distributional term representations: an experimental comparison," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 2004, pp. 615–624.

[22] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." In *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[23] M. P. Villegas, M. Jos´e, G. Ucelay, J. P. Fern´andez, M. A.´Alvarez-Carmona, M. L. Errecalde, and L. C. Cagnina, "Vector-based word representations for sentiment analysis: a comparative study," pp. 785–793.

[24] M. Giatsoglou, M. G. Vozalis, K. Diamantaras, A. Vakali, G. Sarigiannidis, and K. C. Chatzisavvas, "Sentiment analysis leveraging emotions and word embeddings," *Expert Systems with Applications*, vol. 69, pp. 214–224, 2017.

[25] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, p. 533, 1986.

[27] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin, "A Neural Probabilistic Language Model," *The Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[28] T. Mikolov, J. Kopecky, L. Burget, O. Glembek *et al.*, "Neural network based language models for highly inflective languages," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 4725–4728.

[29] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493– 2537, 2011.

[30] E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng, "Improving word representations via global context and multiple word prototypes," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, 2012, pp. 873–882.

[31] A. Mnih and G. E. Hinton, "A scalable hierarchical distributed language model," in *Advances in neural information processing systems*, 2009, pp. 1081–1088.

[32] Y. Li, Q. Pan, T. Yang, S. Wang, J. Tang, and E. Cambria, "Learning word representations for sentiment analysis," *Cognitive Computation*, vol. 9, no. 6, pp. 843–851, 2017.

[33] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato, "Learning longer memory in recurrent neural networks," *arXiv preprint arXiv:1412.7753*, pp. 1–9, 2014.

[34] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent Neural Network Regularization," *Iclr*, no. 2013, pp. 1–8, 2014.

[35] M. Huang, Q. Qian, and X. Zhu, "Encoding syntactic knowledge in neural networks for sentiment classification," *ACM Transactions on Information Systems*, vol. 35, no. 3, 2017, cited By 0.

[36] D. Wu and M. Chi, "Long short-term memory with quadratic connections in recursive neural networks for representing compositional semantics," *IEEE Access*, vol. 5, pp. 16077–16083, 2017, cited By 0.

[37] E. Cambria, D. Das, S. Bandyopadhyay, and A. Feraco, "Affective computing and sentiment analysis," in *A Practical Guide to Sentiment Analysis*. Springer, 2017, pp. 1–10.

[38] X. Fan, X. X. Li, F. Du, and X. X. Li, "Apply Word Vectors for Sentiment Analysis of APP Reviews," no. Icsai, pp. 1062–1066, 2016.

[39] J. H. Lau and T. Baldwin, "An empirical evaluation of doc2vec with practical insights into document embedding generation," *arXiv preprint arXiv:1607.05368*, 2016.

[40] A. Tripathy, A. Agrawal, and S. K. Rath, "Classification of sentiment reviews using n-gram machine learning approach," *Expert Systems with Applications*, vol. 57, pp. 117–126, 2016.

[41] L.-C. Yu, J. Wang, K. R. Lai, and X. Zhang, "Predicting Valence-Arousal Ratings of Words using a Weighted Graph Method," *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL-2015)*, pp. 788–793, 2015.

[42] H. Wu, Y. Hu, H. Li, and E. Chen, "A New Approach to Query Segmentation for Relevance Ranking in Web Search," *Inf. Retr.*, vol. 18, no. 1, pp. 26–50, 2015.

[43] J. Zhang, C. Zong, and Others, "Deep Neural Networks in Machine Translation: An Overview," *IEEE Intelligent Systems*, vol. 15, 2015.

[44] S. Lee, X. Jin, and W. Kim, "Sentiment Classification for Unlabeled Dataset using Doc2Vec with JST," pp. 1–5, 2015.

[45] B. Agarwal, N. Mittal, P. Bansal, and S. Garg, "Sentiment Analysis Using Common-Sense and Context Information," *Computational Intelligence and Neuroscience*, vol. 2015, pp. 1–9, 2015.

[46] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures on Human Language Technologies*, vol. 5, no. 1, pp. 1–167, 2012.

[47] ——, "Sentiment analysis and subjectivity," *Handbook of natural language processing*, vol. 2, pp. 627–666, 2010.

[48] S. Baccianella, A. Esuli, and F. Sebastiani, "SWN 3.0: An enhanced lexical resource for sentiment analysis and opinion mining." in *LREC*, vol. 10, 2010, pp. 2200–2204.

[49] A. Esuli and F. Sebastiani, "SWN: a high-coverage lexical resource for opinion mining," *Evaluation*, pp. 1–26, 2007.

[50] 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*. ACM, 2004, pp. 168–177.

[51] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150.