Information Processing and Management 56 (2019) 1245–1259

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

Deep learning-based sentiment classification of evaluative text based on Multi-feature fusion

Asad Abdi[a,*], Siti Mariyam Shamsuddin[a], Shafaatunnur Hasan[a], Jalil Piran[b]

[a] UTM Big Data Centre (BDC), Universiti Teknologi Malaysia, Johor, Malaysia
[b] Department of Computer Science and Engineering, Sejong University, Seoul, South Korea

ARTICLE INFO

Keywords:
Deep learning
Sentiment analysis
Natural language processing
Neural network

ABSTRACT

Sentiment analysis concerns the study of opinions expressed in a text. Due to the huge amount of reviews, sentiment analysis plays a basic role to extract significant information and overall sentiment orientation of reviews. In this paper, we present a deep-learning-based method to classify a user's opinion expressed in reviews (*called* RNSA).

To the best of our knowledge, a deep learning-based method in which a unified feature set which is representative of word embedding, sentiment knowledge, sentiment shifter rules, statistical and linguistic knowledge, has not been thoroughly studied for a sentiment analysis. The RNSA employs the Recurrent Neural Network (RNN) which is composed by Long Short-Term Memory (LSTM) to take advantage of sequential processing and overcome several flaws in traditional methods, where order and information about the word are vanished. Furthermore, it uses sentiment knowledge, sentiment shifter rules and multiple strategies to overcome the following drawbacks: *words with similar semantic context but opposite sentiment polarity; contextual polarity; sentence types; word coverage limit of an individual lexicon; word sense variations*. To verify the effectiveness of our work, we conduct sentence-level sentiment classification on large-scale review datasets. We obtained encouraging result. Experimental results show that (1) feature vectors in terms of (a) statistical, linguistic and sentiment knowledge, (b) sentiment shifter rules and (c) word-embedding can improve the classification accuracy of sentence-level sentiment analysis; (2) our method that learns from this unified feature set can obtain significant performance than one that learns from a feature subset; (3) our neural model yields superior performance improvements in comparison with other well-known approaches in the literature.

1. Introduction

In recent years, with the rapid development of social media, a vast amount of reviews, opinions and feedbacks are constantly produced from all over the world every day. Many organizations and peoples follow user opinions and comments to decide the quality and performance of a product or service. Therefore, there is a need to analyze these data in order to extract and detect relevant information and the polarized opinion, respectively. On the other hands, it is important for companies and peoples to automatically identify a user's opinion whether it is positive or negative. Opinion/ sentiment analysis is a technique that can be used to determine and classify people's opinion according to their polarity. Sentiment analysis is a task to find automatically subjective

* Corresponding author.
E-mail addresses: seyedasadollah-pd@utm.my (A. Abdi), mariyam@utm.my (S.M. Shamsuddin), shafaatunnur@utm.my (S. Hasan), piran@sejong.ac.kr (J. Piran).

https://doi.org/10.1016/j.ipm.2019.02.018
Received 6 September 2018; Received in revised form 24 January 2019; Accepted 28 February 2019
Available online 20 March 2019
0306-4573/

information of a text such as opinions, sentiments, evaluations, etc. (Abdi, Shamsuddin, Hasan, & Piran, 2018; Rojas-Barahona, 2016). In other words, one of the key role tasks of sentiment analysis deals with the problem of identifying and classifying the polarity in a user-produced content. It is helpful when a user/company wants to know whether a certain opinion is positive or negative.

In the present time, due to the tremendous information available electronically, we need a new technology/mechanism to (1) tackle the overloading of information; (2) obtain the information fast and most efficiently; (3) extract the most relevant and vital information; (4) sift vast volumes of information. However, sentiment analysis technique can be considered as a mechanism to tackle the aforementioned problems. It also helps users to quickly find needed information. Sentiment analysis has been widely investigated in recent years and because of increasing information, it is attracting more attention among the researchers.

In the last years, the huge amount of reviews makes it extremely difficult for users to read and analyse public opinions. Thus, it is necessary to mine the opinions within large volumes of opinionated text into an easily understandable form. We expect a sentiment analysis model to aid users in this problem. In other words, a sentiment analysis model must be able to identify the representative opinion sentence and provides informative information. It is not only helpful for peoples, but also is useful for companies to examine and assess the opinion of costumers in order to collect their opinions which can assist companies in the decision-making process. The sentiment analysis is performed to classify the polarity of a given text, whether the expressed opinion in a text is positive, negative or neutral. Sentiment analysis is the essential task for Natural Language Processing (NLP) with many applications such as web mining, text mining and data mining. The goal of natural language processing is to process text using the computational linguistics, text analysis, machine learning, statistical and linguistics knowledge in order to analyze and extract significant information (Gupta, Tiwari, & Robert, 2016). In the past decades, several methods have been proposed for sentiment analysis. Most of those methods are based on the computational linguistic approach and machine learning approach such as Naive Bayes (NB) and Support Vector Machines (SVMs). Although both of them obtained a good result, several methods in literature have shown that machine learning approach exhibit higher performance than computational linguistic approach.

However, in more recent years, the use of a new technique known as deep learning has attracted the attention of researchers, because it has obtained remarkable results for various NLP tasks (Cliche, 2017). Deep Learning is a part of machine learning architecture including multiple layers of perceptron inspired by the human brain (Du, Cai, Wang, & Zhang, 2016). In other words, deep learning is a model that contains many layers of nonlinear information processing and a method for learning of feature representations at successive layers (Schmidhuber, 2015). There are various deep learning models such as deep neural networks (DNN), convolution neural networks (CNN), deep Restricted Boltzmann Machine (RBM), etc. However, while the use of deep learning model in NLP has illustrated very desirable results for many tasks, it seems that they have not yet reached the level to overcome the existing problems and there is room for improvement.

In this paper our objective is 2-fold. *First*, we propose a deep learning-based method to classify a user's opinion expressed in reviews using multi-feature fusion (*called* RNSA). The RNSA employs a deep learning-based method, RNN-LSTM, for sentiment analysis at the sentence level. It takes the word embedding, sentiment and linguistics knowledge features as the input of RNN-LSTM. Consequently, the encoded feature from RNN-LSTM is considered as the sentence level representation. To the best of our knowledge, a deep learning-based method in which a unified feature set which is representative of word embedding, sentiment knowledge, sentiment shifter rules, statistical and linguistic knowledge, has not been thoroughly studied for a sentiment analysis. We incorporate these features in order to overcome the following drawbacks: (a) words with similar semantic context but opposite sentiment polarity; (b) contextual polarity; (c) sentence types; (d) word coverage limit of an individual lexicon; (e) word sense variations. We describe each of them in the following paragraphs.

Second, we aim to compare the performance of the proposed method with the other existing systems.

The RNSA aims to cope with the following drawbacks which can be summarized as follows:

(1) Recently, many deep learning approaches in natural language processing exploits word embedding learning algorithm as a popular method for vector representation of each word or sentence. They usually use the freely available word2vec[1] vector that was trained on 100 billion words from Google News. However, one of the important drawbacks of word embedding model is ignoring the sentiment polarity of the words (Araque, Corcuera-Platas, Sanchez-Rada, & Iglesias, 2017; Giatsoglou et al., 2017). As a result, words with opposite polarity are mapped into close vectors. For instance, the words "*like*" and "*dislike*" can appear in a similar context such as "*I like the movie*" and "*I dislike the movie*". With respect to their syntactic structure and word co-occurrences both words have similar vector representations, but from a sentiment point of view, their vector representation must be different as they are of opposite polarities. Thus, a vector representation based on the word embedding algorithm has not enough sentiment information to perform sentiment analysis and is not able to precisely capture the overall sentiment of a sentence. To overcome the aforementioned problem, the prior sentiment knowledge information is incorporated into the word embedding as word representation. Prior sentiment knowledge conveys complementary information that is not available in word co-occurrence and, hence, can enrich word embedding for sentiment analysis. In sentiment analysis, sentiment lexicons are valuable resources that can provide prior knowledge. A sentiment lexicon includes a sentiment score of a word based on its positivity and negativity that can distinguish the sentiment polarity of a word. The current stage also combined several sentiment dictionaries (a) in order to tackle the word coverage limit; (b) *o*n the other hand, various sentiment dictionaries complement each other.

(2) In addition, the word embedding model cannot differentiate the senses of a word and creates a single representation per word form. For example, the word vector for "*apple*" as a fruit is equal to its word vector as a company. Therefore, the current stage also applies a strategy to solve this problem.

(3) Contextual polarity is one of the most challenging tasks in sentiment analysis. It refers to context-based sentiment analysis, where the word's prior polarity changes with respect to different context. The contextual issue indicates: (a) the problem of negations

which may appear in different places in a sentence. If we consider an example, "*I do not like this movie*", here the negation word, "*do not*" changes the polarity of the word "*Like*". (b) specific particles such as "*but*", "*despite*", etc., that affect the sentiment analysis outcome. As an example, given the sentence "*the car is good-looking but very expensive*". The but-clause changes the polarity of the previous phrase "*the car is good-looking*".

(4) On the other hand, since the performance of a sentiment analysis method relies on the different types of sentences (*e.g., subjective sentences, comparative sentences, conditional/question sentences, etc.*) (Chen, Xu, He, & Wang, 2017) and most of the existing methods ignore different sentence types, we consider the various types of sentences in sentence-level sentiment analysis. However, we integrate several strategies to tackle the above-mentioned problems: contextual polarity and types of sentences.

In summary, the contributions of the present work can be summarized as follows:

(1) To the best of our knowledge, a deep learning-based method in which a unified feature set which is representative of word embedding, sentiment knowledge, sentiment shifter rules, statistical and linguistic knowledge, has not been thoroughly studied for a sentiment analysis.
(2) We also integrate multiple strategies to handle: (a) types of sentences; (b) contextual polarity; (c) word sense variations; (d) sentiment shifter rules; (e) integration of sentiment information and word embeddings; (f) sentiment score calculation; (g) word order information and semantic relationships between words, which enable our method to achieve superior performance.
(3) The method encodes several valuable resource-information latent in a sentence to (a) generate augmented vector; (b) learn a better sentence representation; (c) improve the method performance. A hybrid vector is constructed for the representation of each sentence using the sentiment-based, word embedding-based, statistical and linguistic knowledge-based feature vectors.
(4) The method integrates several sentiment lexicons in order to tackle the word coverage limit. On the other hand, various sentiment dictionaries complement each other.
(5) Finally, we conduct extensive experiments to show the effectiveness of our proposed method. Compared with the existing methods and experiment results, the method achieves superior performance on all the measure metrics and the significant results affirm the suitability of the proposed method.

This paper is structured as follows. We summarize the recent related works in the next section. Our proposed method, RNSA, is presented in detail in Section 3. Our experimental results are explained in Section 4. Finally, we provide conclusions and ideas for future research.

2. Related work

2.1. Sentiment analysis

With the huge amount of user-generated texts, extraction of significant information from numerous documents has gained much attention from the community of Natural Language Processing (NLP). Sentiment analysis (*also known as Opinion mining*) is an active research area of NLP that aims to identify subjective information and determine the sentiment orientation (*e.g., positive or negative*) of a given text (Abdi, Shamsuddin, & Aliguliyev, 2018a; Sun, Li, & Ren, 2016). In other words, sentiment analysis is the computational study of people's opinions, which aids users to gather desirable information for decision making. Recently, sentiment analysis has been applied to various domains such as commercial, political and social media.

In general, sentiment analysis can be divided into three levels (Ain et al., 2017; Rana & Cheah, 2016): (1) sentence level; (2) document level; (3) aspect-based level. At the document level, the sentiment analysis process determines the overall sentiment polarities of the given document. It assumes that the document discusses only one topic or a single entity. Similar to document level sentiment analysis, the task of the sentence level is to determine whether a sentence expresses a positive, negative or neutral opinion. This level also examines the subjectivity classification, which aims to determine whether a sentence is an objective (*not opinionated*) or subjective (*opinionated*). Compared with document and sentence level sentiment analysis, aspect level sentiment analysis is employed to acquire details of the opinionated text. In other words, it extracts the people's opinion expressed on the entity and the feature/attribute of an entity. The task of aspect-based level sentiment analysis includes three main steps: entity/object identification, feature/attribute extraction and attribute polarity identification.

Currently, the most sentiment analysis method can be divided into three categories (Sun, Luo, & Chen, 2017; Vateekul & Koomsubha, 2016): lexicon-based sentiment analysis, machine learning-based sentiment analysis and the hybrid approach. A dictionary-based method determines the polarity of a text based on the total sum of comprised positive or negative sentiment words in a text (*e.g.*, Yadav & Chatterjee, 2016). A sentiment lexicon includes a set of negative and positive values assigned to corresponding words and phrases. In comparison with machine learning-based method, the lexicon-based method is a suitable method for sentiment analysis, since it needs fewer resources and does not need some annotated corpora. Furthermore, a sentiment lexicon can also be used in a machine learning-based approach in order to make sentiment related features. However, a lexicon-based approach suffers from word coverage of sentiment words in a text. Compared with the lexicon-based approach, a machine learning-based approach uses the most common machine learning algorithms to classify sentiment orientation. In this type of method, feature definition/extraction is very important for a learning-based method. Many studies used the most common standard classifier for sentiment analysis such as Support Vector Machine (SVM) or Naive Bayes (Abdi, Shamsuddin, Hasan, & Piran, 2018b). Finally, in a hybrid approach, in order to improve the accuracy of a text classification system, the dictionary-based methods were combined with machine learning-based

approaches. Recently, deep learning has also been applied to sentiment analysis and achieved promising results (Rezaeinia, Ghodsi, & Rahmani, 2017; Sun et al., 2017).

Deep learning is a new area of machine learning to simulate the learning functions of the human brain. In other words, deep learning is a neural network (NN) with multiple hidden layers. A neural network contains a various number of nodes per layer and hidden layers between the input and output layers. However, given an input, a NN approach is able to learn features and to produce a classification result. Some researchers found out that the traditional sentiment analysis approaches are insufficient, and the results are not very satisfied (Day & Lin, 2017). Hence, a deep learning model has achieved significant attentions in various NLP tasks. Several deep learning-based methods have been proposed for sentiment analysis tasks such as Caliskan, Bryson, and Narayanan (2017), Chen, Xu, He, Xia, and Wang (2016) and Liu, Liu, Shan, and Wang (2018).

Furthermore, there are two types of opinions: (1) regular opinion that can be categorized into direct and indirect opinion. In direct opinion, a user expresses his/her opinion/feeling directly on an attribute, while in indirect opinion, an opinion is expressed indirectly on an attribute. (2) The comparative opinion which expresses the difference between entities. Moreover, a review text can be an explicit review or implicit review text. An explicit review is a subjective representation that demonstrates regular or comparative opinion, while the implicit review is an objective representation that indicates regular or comparative sentiments. A subjective text expresses subjective views and feelings, while an objective text expresses factual information, no sentiment or opinion. Emotion is also related to sentiment analysis that presents our feeling and opinion. The researches categorized people's opinions into some groups such as, "*love*", "*joy*", "*surprise*", "*anger*", "*sadness*", and "*fear*".

Deshwal and Sharma (2016) introduced a supervised sentiment classification method based on the set of features like emoticons, exclamation and question mark symbol, unigrams. They also compared the performance of various supervised classification method in order to find a well-known method for increasing the efficiency of sentiment classification system. The experimental results displayed that discriminative multinomial Bayes and sequential minimal optimization approach produced improved overall results.

The method proposed by Tripathy, Agrawal, and Rath (2016) classifies the reviews into either positive or negative polarity using n-gram techniques *i.e.*, unigram, bigram, trigram and the combination of them. They considered four different supervised machine learning methods (*e.g.*, Naive Bayes (NB), Maximum Entropy (ME) and Support Vector Machine (SVM)) to classify these reviews. They also examined the performance of the aforementioned machine learning approaches in terms of precision, recall, *f*-measure, and accuracy. The result obtained by the proposed method is better than the results produced by the existing methods. On the other hands, it is observed the unigram and bigram increase the accuracy of the method and the tri-gram, four-gram and five-gram decreases the accuracy.

Mohammad, Kiritchenko, and Zhu (2013) developed a method based on the Support Vector Machine (SVM) to classify tweet using a set of features including *'word n-grams', 'all-caps', 'part-of-speech (POS)', 'punctuation', 'elongated', etc*. Miura, Sakaki, Hattori, and Ohkuma (2014) also proposed a sentiment classification system based on the supervised machine learning approach, Logistic Regression algorithm. They used the groups of features that have used by Mohammad et al. (2013), such as *'character n-grams, lexicons, 'word senses', etc*. the basic linguistic functions like a spelling corrector and a word sense disambiguator were also applied in the text pre-processing. Finally, Amir, Almeida, Martins, Filgueiras, and Silva (2014) also introduced a method to assign a class label to a tweet based on the Logistic Regression algorithm the three groups of features: *'word-based, 'lexicon', and 'syntactic'*. Bermingham and Smeaton (2010) also applied two supervised learning approaches, *i.e.*, SVM and Naive Bayes, to classify the sentiments of tweets. The binary feature was used in their method.

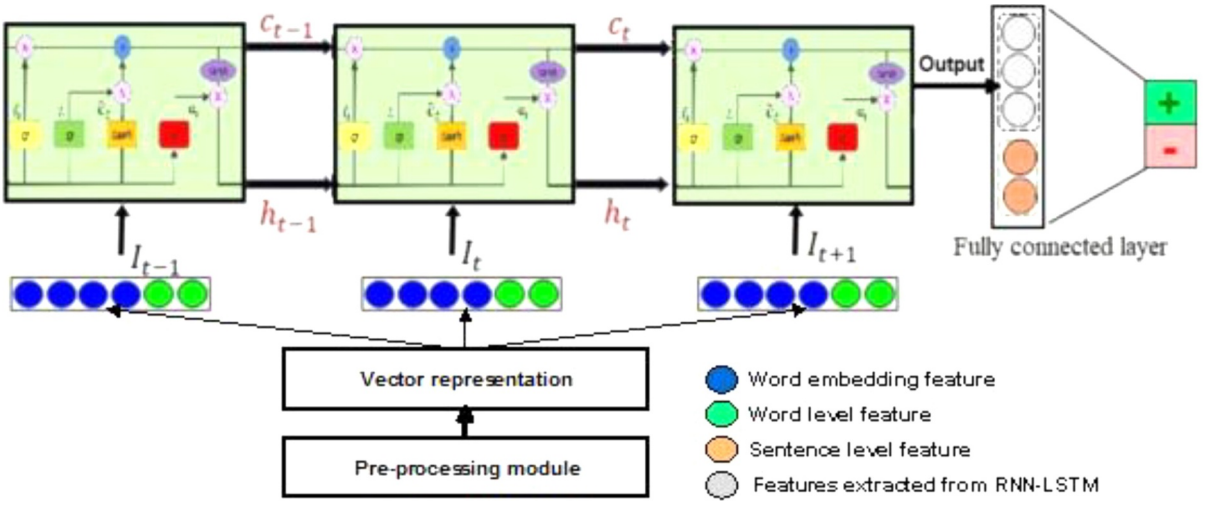

Fig. 1. The architecture of the proposed method RNSA.

3. Proposed method

In this paper, we present a method for sentiment analysis. It includes two main parts as shown in Fig. 1. The first part involves the *pre-processing module*. The second part consists of *sentiment analysis*.

(1) *Pre-processing module:* It includes the basic linguistic functions. The output of the pre-processing will be the set of hand-crafted features and word embedding representation.
(2) *Sentiment analysis:* It employs the RNN-LSTM algorithm for the sentence-level classification task.

A detailed description of these levels is provided in the following sections.

3.1. Pre-processing

Before applying the RNSA method, we perform data pre-processing. It allows to produce a higher quality of opinion summary and reduce the computational complexity. Briefly, pre-processing contains the following tasks:

The raw text is first given as input to the pre-processing step. In this step, each document is decomposed into several paragraphs. Subsequently, the paragraphs are further decomposed into sentences. Secondly, the tokenization, a basic approach of the text pre-processing, splits the sentences into words contained in it. The sequence of tokens is also passed to the stop-word removal procedure to eliminate words which do not convey any meaning and are useless for text mining. The stop-word[1] includes words like articles, conjunctions, prepositions, pronouns which are high-frequency words and provide little useful information. We also employ the stemming procedure to reduce a word to its root form. Part of Speech (POS) tagging is also used to classify the words of text on the basis of part of speech category (‘*noun*’, ‘*verbs*’, ‘*adverb*’, ‘*adjectives*’) they belong. The POS tagging provides useful lexical information.

3.1.1. Sentiment shifter rules

the pre-processing step also applies sentiment shifter rules on datasets. The sentiment shifter refers to the context-based sentiment analysis. It includes some words like *‘but’, ‘while’, ‘however’, etc.* that can change the sentiment orientation of the sentence following them (Xia, Xu, Yu, Qi, & Cambria, 2016). If we consider an instance, “*the car is good-looking but very expensive*”, the word ‘*but’* changes the polarity of the sentence “*the car is good-looking”*. In other words, the polarity of sub-sentence before the word ‘*but’* and after it is opposite to each other. Therefore, the overall opinion is only in the sub-sentence following the word ‘*but’*. Hence, the sub-sentence before the word ‘*but’* is unessential part and can be removed. However, the aforementioned pitfall leads to the potential mis-classification in sentiment analysis. Thus, a set of rule strategy is required to deal with the specific particles (*e.g., ‘but’, ‘while’, ‘however’, etc.*) that can affect the result of a sentiment analysis. In this work, we used the rules proposed by Nguyen and Nguyen (2017) and Xie et al. (2014), as shown in Table 1.

3.1.2. Sentence vector representation

a popular approach for text representation is bag-of-word (BOW) approach, a sentence is represented using a set of its words where each word is weighted using various methods (*i.e., term-frequency, term- frequency and inverse document frequency, True/False, 0/1*). The BOW approach suffers two main issues (Ahmad et al., 2015; Al-Sallab et al., 2017; Xiao et al., 2018): (1) a vector representation based on the BOW is sparse since each sentence only includes a small number of words. A sparse representation can cause a problem for the training process since some words may only appear in the testing dataset and never be seen in the training dataset. (2) In addition, the BOW is not able the distinguish the meaning of two sentences because it ignores the syntactic structure and word order in sentences. In other words, different sentences, can have the same vector representations. The *n*-gram is another popular method that performs best (Joachims, 1998). It considers the word order in a sentence, but it also suffers from data sparsity and high dimensionality.

Recently, in order to tackle the aforementioned problems, most of the proposed systems used the word embedding technique to produce a low-dimensional vector for sentence representation. Due to its ability to encode syntactic and semantic properties of words, the word embedding technique has been used for various NLP tasks such as parsing (Bansal, Gimpel, & Livescu, 2014), POS tagging (Lin, Ammar, Dyer, & Levin, 2015), etc. Word2vec[2] is one of the popular word-embedding models which is used to perform the computation of the word vector representations. It includes Continuous Bag-of-Words model (CBOW) (Mikolov, Chen, Corrado, & Dean, 2013a), and Skip-Gram model (SG) (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013b) to provide high-quality word embedding vectors. The CBOW predicts the target word based on the embeddings of its context words, while the SG predicts the surrounding words given the target word. We also employ the word2vec approach in our proposed method and consider several strategies to cope with its drawbacks.

3.2. Sentiment analysis

The sentiment analysis as shown in Fig. 1, includes the following parts: input vectors, RNN-LSTM layer, concatenation layer and

[1] http://xpo6.com/list- of- English- stop- words.
[2] https://code.google.com/archive/p/word2vec/.

Table 1
Sentiment shifter rules for text sentiment analysis.

Example	Semantic rules
"I am sorry $_{\text{sub-sentence1}}$, but I can't pay you $_{\text{sub-sentence2}}$"	If a sentence includes "*but*", ignore the sentiment orientation of sub-sentence$_1$ and only consider the sentiment of the sub-sentence$_2$.
"He played well $_{\text{sub-sentence1}}$, despite being injured $_{\text{sub-sentence2}}$	If a sentence includes "*despite*", only consider the sentiment of the sub-sentence$_2$.
"I wouldn't wear those shoes $_{\text{main clause}}$ unless I was trying to break my ankle $_{\text{subordinate clause}}$"	If a sentence includes" unless", and the" unless" is followed by a negative clause, ignore the" unless" clause.
"My throat is killing me, and while I got a decent night's sleep last night, I still feel like I'm about to fall over"	If a sentence includes "*while*", ignore the sentence following the "*while*" and consider the sentiment only of the sentence that follows the one after the "*while*"
"I am glad I got the job; however, I'll have to travel more often"	If a sentence includes "*however*", ignore the sentence before "*however*" and consider the sentiment of the sentence after "*however*"

fully connected layer. In input vectors, sentences are firstly considered via the pre-processing step to standardize sentences and capture only important information containing the sentiment of a sentence. Therefore, the integration of word-level embedding, sentiment and linguistic knowledge features (*word-level feature*) are fed into RNN-LSTM layer to generate a sentence-wide feature set. In the concatenation layer, we augment the extracted vector representation from last LSTM cell with the sentiment and linguistic knowledge features (*sentence-level feature*) to form a final vector representation for each sentence. This vector is taken to the fully connected layer, then the '*sigmoid*' function reveals the sentiment label, subjective (*positive/negative*). We explain in detail the kinds of RNN and LSTM that we use as follows:

Some background on RNN — The RNN is a popular neural network algorithm that has been used in many NLP tasks. RNN is able to model variable length input sequences and learn long-term dependencies among the sequence (Schmidhuber, 2015; Zhang, Yang, Chen, & Li, 2018). In other words, the ability to capture contextual information in sequential data can be useful to obtain semantics of long text. Fig. 2 shows a recurrent neural network architecture and the connections between the hidden units. The hidden unit of an RNN is equal to $h_t = f(X_t U_t + h_{t-1} W_{h_{t-1} h_t})$, where t is time step, W, U, V are the weight matrix and f is an active function. In an RNN model, each word (*hidden unit*) affects all the subsequent words (*hidden units*). Given the sentence "*I like movie*", the RNN processes the sentence as follows: at the first step ($t = 0$), it processes the word "*I*" and computes the hidden unit h_0 ($h_0 = f(X_0 U_0)$). At second step ($t = 1$), the word "*like*" is processed and the h_0 also fed into hidden unit $h_1 (h_{the\ 1} = f(X_1 U_1 + h_0 W_{01}))$. Finally, it processes the word "*movie*" and feeds the h_1 into h_2 ($h_2 = f(X_2 U_2 + h_1 W_{12})$).

However, word order information and semantic relationships between words have an important impact on sentiment classification performance. Since the RNN model deals with time-series data and can learn from a sequence of words, it can solve the problem of the negation which may appear in different places in a sentence. If a negation word (*e.g., 'not', 'never', etc.*) appears in a sentence, the hidden unit of this word will affect the polarity/ emotion of the subsequent words, and as result, the polarity of the sentence. For instance, the words '*bad*' and '*not*' are negative, but the phrase '*not bad*' which is combined of these two words has a positive meaning.

The RNN suffers from vanishing gradients (Bengio, Simard, & Frasconi, 1994); hence, (1) the length of the sequences that an RNN can process is limited and (2) makes it difficult to learn long-distance correlation in sequence. Therefore, LSTM network was proposed to tackle the long-distance dependencies problem of RNN using a memory cell that preserves state over long periods of time (Hochreiter & Schmidhuber, 1997). The core of LSTM is a memory cell which four main gates regulate the information flow into and out of the cell: the input gate, output gate, forgetting gate and candidate memory cell (Fig. 3). The cell decides what to keep in and what information to erase from memory via gates. A LSTM architecture is not fundamentally different from RNN, but it employs different functions to calculate the hidden state. The LSTM memory cell is updated using the following steps:

Working of gates — The gates are calculated at every timestep t using the following equations:

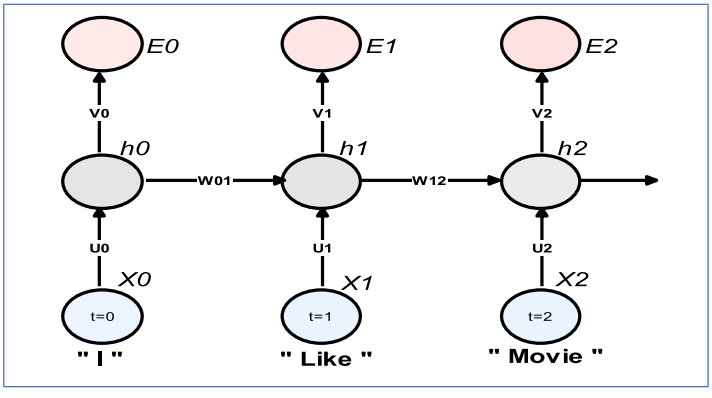

Fig. 2. The architecture of a Recurrent Neural Network (RNN).

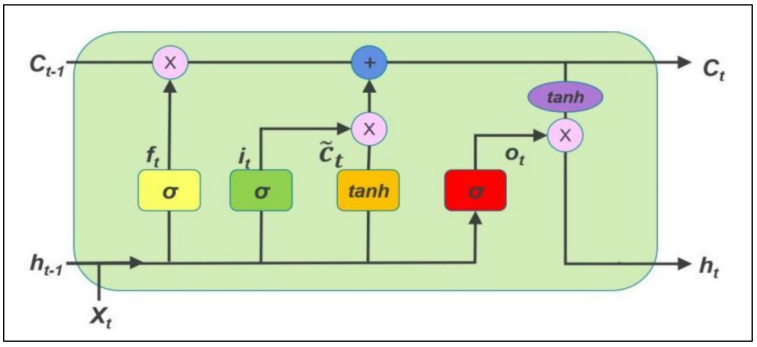

Fig. 3. The LSTM architecture.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{2}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3}$$

$$\tilde{c}_t = \mathrm{Tan}h(W_c x_t + U_c h_{t-1} + b_c) \tag{4}$$

where h_{t-1} is the hidden unit at time step $t-1$, x_t is the input at time t. b is bias vector and subscript i, f, o and c indicate the input gate, forget gate, output gate and cell state, respectively. U and W indicate the weight matrix of each gate. i_t, f_t, o_t and $\tilde{c}_t$ represent the input gate, forget gate, output gate and candidate memory cell state, respectively. Tan$h$ and σ are hyperbolic tangent and sigmoid function, respectively.

Memory cell update — In this step the cell state at time t is updated using the Eq. (5). The c_{t-1} is multiplied by f_t, then the result of $(i_t \times \tilde{c}_t)$ is added.

$$c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t \tag{5}$$

where c denotes the variable for cell state.

Hidden layer output — Given the new state of the memory cells, c_t, and o_t the hidden state, h_t is computed as follows:

$$h_t = o_t \times \mathrm{Tan}h(c_t) \tag{6}$$

However, since the RNN can process sequential data and the LSTM can control the flow of information, cope the vanishing gradient and aid the RNN to capture long-term dependencies, we employ RNN-LSTM in sentiment analysis process.

3.2.1. Input features

A feature is a characteristic of a text for capturing patterns in data. Feature extraction process transforms the input dataset into a set of features in order to improve the overall quality of text classification. The procedure of feature extraction is explained in detail as follows:

Word embeddings: A neural network needs a vector representation of each word or sentence as an input to the network. Word embedding is a method that can be used for feature learning. It is able to transform words into real-valued and low-dimensional vectors and capture useful syntactic and semantic properties about the words. We use word2vec to translate a sentence into its vector representation. The word2vec was trained on 100 billion words from Google News. In our method, given a sentence $S = \{w_1, w_2, ..., w_n\}$, where n indicates the number of words in S. Each word, w_i, is associated with a D-dimensional vector embedding, $X_i \in R^d$. The sentence of length n is represented as follows: All the word vector representations are concatenated in their respective order, $X_{1:n} = X_1 \oplus X_2 \oplus ... \oplus X_n$, where $\oplus$ is the concatenation operator. Furthermore, each sentence is padded with zero-vectors to a fixed length.

Sentiment and linguistic knowledge: Although the word embedding learning method has been applied to various NLP tasks, it is not effective enough for sentiment analysis. The word embedding approach has some obvious drawbacks which can be summarized as follows:

Word-level features:

Part-of-speech (POS) tagging — Word-sense disambiguation (*when a word has multiple meanings*) is one of the problems with the word embedding approach. It cannot distinguish the senses of words and creates a single representation per word form (Kamkarhaghighi & Makrehchi, 2017). For example, given two sentences (*"The coach devised a great play", "The boy went out to play in the yard"*), each sentence associates a different meaning of the word *'play'* based on the context of the word's usage in a sentence. However, in order to cope the current problem, we use a simple approach, POS tagging, as additional feature for each word: (*"The/DT children/NNS went/VBD out/RP to/TO play/VB in/IN the/DT park/NN", "The/DT coach/NN devised/VBD a/DT great/JJ play/NN"*). We use a constant binary vector (six-dimensional vector: '*noun*', '*verb*', '*adjective*', '*adverb*', '*preposition*', '*conjunction*') and concatenate with the corresponding word embedding vector.

Sentiment-encoded word embedding — The most serious problem with the word embedding approach is that the approach ignores the sentiment information of a word. as consequence, two words with similar contexts and opposite polarity (*e.g.*, *'poor' and 'nice'* in *"He is a nice guy"* and "*He is a poor guy*") are mapped into close vectors in the embedding space. From a sentiment point of view, the vector representations of both above-mentioned words should be very different as they convey opposite polarity. However, given a vector representation that has insufficient sentiment information, the RNSA is not able to accurately obtain the overall sentiment of the sentence, therefore, it affects the sentiment classification performance. In order to tackle the current problem, we encode the sentiment information into the word embedding process to learn better representation for sentiment analysis. To do this, we use a 2-dimensional binary vector of prior sentiment polarity of words (positive/negative polarity). For each word of the input sentence, the corresponding sentiment polarity is provided by the prior knowledge/ external source such as a sentiment dictionary. One problem that we will encounter is that a word of the sentence is not in the lexicon. In such a case, we employ the SSM method as described in section "(b) *Semantic Sentiment Method (SSM)*". However, the binary vector is appended to the end of the corresponding word embedding. In sentiment analysis, a sentiment lexicon can provide prior knowledge. A sentiment lexicon contains a set of words and their corresponding sentiment score/polarity. In our work, we combine ten sentiment dictionaries in order to tackle the word coverage limit of an individual sentiment dictionary (*called MSD*). We explain the combination process in the following section:

(a). Sentiment lexicons combination: This section explains the creation of MSD. We integrate several sentiment dictionaries. The existing dictionaries with different size and format are mapped into three categories as follows. Since the lexicons have various formats, we first standardize them, then the score of each word in MSD is computed by averaging sentiment scores of overlapping words.

Category 1: a sentiment dictionary includes sentiment score with various numeric ranges (*e.g.*, (Nielsen, 2011), $[-5, +5]$; (Mohammad et al., 2013), $[-7, +7]$; (Cambria, Poria, Bajpai, & Schuller, 2016), $[-1, +1]$; (Taboada, Brooke, Tofiloski, Voll, & Stede, 2011), $[-5, +5]$; (Baccianella, Esuli, & Sebastiani, 2010), [0, 1]. For this type of lexicons, a sentiment score is normalized from $[-5, +5]$/ $[-7, +7]$/ to $[-1, +1]$. Regarding Baccianella et al. (2010), each row of this lexicon includes a synset with POS information, an ID that maps the synset to WordNet, the positive (Pos)/negative (Neg) sentiment scores and a gloss which contains the meaning and a sample usage of the terms present in the synset. Each synset is a group of terms that are synonyms of one another. A synset is '*objective*' if the following equation is equal to 1, $ObjScore = 1 - (PosScore + NegScore)$. Eq. (7) (Khan, Qamar, & Bashir, 2016) is used to calculate the score of each word within the range of $[-1, 1]$:

$$Senti\_score = (Positive_{score} - Negative_{score}) \tag{7}$$

If *senti_score* is greater than zero, less than zero or equal zero the sentiment word orientation is positive, negative or neutral/ objective, respectively.

Category 2: The sentiment dictionary classifies the words into '*positive*', '*negative*' and '*neutral*' (*e.g.*, Hu and Liu, 2004 and Riloff and Wiebe, 2003). In this type of category, we assign $+1$, -1, 0 to positive, negative and neutral words, respectively.

Category 3: The sentiment dictionary classifies the words into several types of emotions such as '*bad*' and *'happy'* or categorize ito several groups such as '*positiv', 'affil', 'strong', 'week', 'fail', 'passive'* (*e.g.*, Strapparava and Valitutti, 2004 and Stone and Hunt, 1963). Therefore, we assign $(+1)$ to words from '*positive-emotion' or 'positive-group', (-1) to 'negative-emotion' or 'negative -group' and (zero/ 0) to 'neutral- emotion'*.

(b). Semantic Sentiment Method (SSM): As mentioned above, the limited words coverage is a major limitation of a sentiment lexicon. However, we exploit the SSM to specify the score of a word (*W*), if it does not exist in MSD. Let $WS = \{sw_1, sw_2 \cdots ..., sw_N\}$ include the synonymous words of the *W*. For each sw_i of *WS,* the following tasks are performed: (1) if the sw_i exists in the MSD, then SSM returns its sentiment score (*SC*); (2) the SSM checks, if the *SC* value is greater than 0 (*zero*), then SSM adds the *SC* to the Pos_{sw}; (3) if the *SC* value is smaller than 0, then SSM adds the *SC* to the Neg_{sw}. Finally, the following equation is used to compute the sentiment score of *W*.

$$\text{Total Sentiment Score } (W) = \frac{\sum Pos_{sw} - \sum Neg_{sw}}{m + n} \tag{8}$$

Let *n* and *m* indicate No. of positive and negative words respectively. ΣPos_{sw} and ΣNeg_{sw} are defined as follows: $\sum Pos_{sw} = \sum SC_{positive}$ and $\sum Neg_{sw} = \sum SC_{negative}$.

Sentence-level features:

Apart from word-level features, there are sentence-level features that cannot be ignored for training the sentence-level. The following features are extracted at sentence level:

Lexicon feature:

- The number of positive sentiment tokens in a sentence.
- The number of negative sentiment tokens in a sentence.

The total sentiment score feature:

- The sum of the scores of the sentiment tokens.

Negation features:

- The frequency of individual negation words within a sentence.

Punctuations feature:

- The frequency of exclamation ('!') mark.
- The frequency of question ('?') mark.

POS feature:

- The frequency of nouns, adjectives, verbs, adverb.

Sentence types:

We also considered the different types of sentences as a feature (Liu, 2012; Chen et al., 2017; Narayanan, Liu, & Choudhary, 2009): (1) subjective/ objective sentence: an objective sentence expresses factual information, while and a subjective sentence expresses an opinion or sentiment; (2) interrogative sentence/conditional sentence: an opinionated sentence may not present any opinion or sentiment such as, "*may you tell me which Samsung laptop is good?*", "*If I can find a good laptop in the shop, I will buy it*" and "*is your bicycle in good condition?*". All sentences contain sentiment words, but they do not express a positive or negative opinion on the laptop. However, all conditional and question sentences do not express opinion or sentiments. We use a constant binary vector (four-dimensional vector: '*objective*', '*subjective*', '*question*', '*conditional*') and concatenate with other features extracted at sentence level.

- Is the sentence subjective?
- Is the sentence objective?
- Is the sentence question/ interrogative sentence?
- Is the sentence conditional sentence?

4. Experimental results and discussion

The important research question here is – *does the RNSA model help to improve sentiment analysis performance?*". Throughout the rest of this section, we address the above-mentioned question in detail. In this section, we conducted experiments to verify the validity of the proposed method as follow.

4.1. The dataset

In order to have an extensive exploration of the RNSA, we conduct the experiments on three public datasets that are freely available from online repositories: (1) the Movie Review (MR) Dataset[3] (Maas et al., 2011): it is a collection of movie-review documents labeled with respect to their overall sentiment polarity (positive or negative). The MR dataset includes of 50,000 binary labeled reviews from IMDB (*positive* 25,000*, negative* 25,000). (2) DUC[4] 2001 and 2002 datasets: The Document Understanding Conferences (DUC) datasets are the commonly used evaluation corpora for summarization. The documents are all from the news domain and are grouped into various thematic clusters. The DUC 2002 dataset includes 567 documents on 59 different topics. The DUC 2001 dataset contains 60 sets of approximately 10 documents. It includes two main tasks: (a) single-document summarization: given a single document, a generic summary of the document with a length of approximately 100 words is created; (b) multi-document summarization: given a set of documents, four generic summaries of the documents with lengths of approximately 400, 200, 100, and 50 words are created.

To examine the performance of our proposed method, we also need a gold standard data, which is a set of all correct results. For this purpose, we employed three annotators, (1) two English teacher with good reading skills and understanding ability in the English language; (2) a lecturer with experience in English language teaching. The annotators aimed to create a gold standard (*including training data and testing data*) using the opinionated sentences. A gold standard is created using the following processes: (1) the annotators split the text contents into sentences, and then tag the opinion of each sentence with *'Polarity = Pos', 'Polarity = Neg', 'Polarity = Neu'* or *'Polarity = null'*. (2) The sentences are categorized into two distinct groups: objective sentences (*factual sentences, a sentence not expressed any opinion, 'Polarity = Neu' or 'Polarity = null'*) and subjective sentences (*'Polarity = Pos', 'Polarity = Neg'*). In our experiment study, all the above data sets are randomly split into training set (70%) and test set (30%).

4.2. Performance measurement

In order to evaluate the performance of RNSA, we used three various standard measures,[5] Precision (denoted as Prec), Recall (denoted as Rec), $\underline{F}$ measure (denoted as $\underline{F}_1$). The precision and recall are computed using the Eqs. (9) and (10). Precision is the ratio

[3] http://www.cs.cornell.edu/people/pabo/movie-review-data/.
[4] http://duc.nist.gov.
[5] https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/.

Table 2
Description of TP, TN, FP, FN.

		Predicted class Class (Yes/+)	Class (No/−)
Actual class	Class (Yes/+) Class (No/−)	TP FP	FN TN

of correctly predicted positive observations to the total predicted positive observations (Manning, Raghavan, & Schütze, 2008). Recall is the ratio of correctly predicted positive observations to the all observations in actual class (Yes/+). These measures can be interpreted as follows. As shown in Table 2, TP means '*True Positive*' – these are the correctly predicted positive values which means that the value of actual class is (Yes/+) and the value of predicted class is also (Yes/+). FP means '*False Positive*' – when actual class is (No/−) and predicted class is (Yes/+). TN, '*True Negatives*' – these are the correctly predicted negative values which means that the value of actual class is (No/−) and value of predicted class is also (No/−). *FN*, '*False Negatives*' – when actual class is (Yes/+) but predicted class in (No/−).

$$Prec = \frac{TP}{TP + FP} \tag{9}$$

$$Rec = \frac{TP}{TP + FN} \tag{10}$$

There is an anti-correlation between precision and recall (Manning et al., 2008). It means the recall drops when the precision rises and vice versa. In other words, a system attempts for recall will get lower precision and a system attempts for precision will get lower recall. To take into consideration the two metrics together, a single measure, called *F*_measure, is used. *F*_measure is a statistical measure that merges both precision and recall. It is calculated as follows:

$$F_1 = \frac{1}{\delta \cdot \frac{1}{P} + (1 - \delta)\frac{1}{R}} = \frac{(\gamma^2 + 1)Prec \cdot Rec}{(\gamma^2)Prec + Rec} \tag{11}$$

where $\gamma^2 = \frac{1-\delta}{\delta}$, $\alpha \in [0,\ 1]$, and $\gamma^2 \in [0,\ \infty]$. If a large value ($\gamma > 1$) assigns to the γ, it indicates that precision has more priority. If a small value ($\gamma < 1$) assigns to the γ, it indicates that recall has more priority. If $\gamma = 1$ the precision and recall are assumed to have equally priority in computing *F*_maesure. *F*_measure for $\gamma = 1$ is computed as follows:

$$F_1 = \frac{2 \cdot Prec \cdot Rec}{Prec + Rec} \tag{12}$$

where *Prec* is precision and *Rec* is recall.

4.3. Sentiment analysis

Each sentence is represented using the word embedding, sentiment and linguistic knowledge features. Then, the RNN-LSTM receives the current feature vector and creates an output. Finally, in the output layer of the RNN-LSTM, the concatenation of feature vector extracted from the last LSTM cell and the sentence-level feature is feed into a fully connected layer. We use '*sigmoid*' function to classify data into two classes. This network is trained by stochastic gradient descent (*the optimization algorithm*) with cross-entropy (*negative* log-*likelihood*) loss functions. We use a learning rate of 0.03 to minimize the loss function. The cross-entropy (Bishop & Bishop, 1995) is one of the common methods for evaluating the loss function. The loss function (*LF*) is computed using the cross-entropy formula as follows: $Lf(y,\ \tilde{y}) = -\sum y_i \ln(\tilde{y})$, where *y* is a gold distribution and $\tilde{y}$ is a predicted distribution (the model output distribution). In addition, in our experiment, we employ l2 constraint regularization and dropout technique (Hinton, Srivastava, Krizhevsky, Sutskever, & Salakhutdinov, 2012) to build a robust system. Dropout is an algorithm for training neural networks to overcome overfitting problem. It prevents co-adaptation of hidden units by randomly dropping units during training (Kim, 2014). The idea behind the dropout method is that during the training phase randomly sets hidden unit values to zero with probability of a p. The proportion of units to be dropped is a hyper-parameter to be determined by the user. We also impose a l2 norm constraint during training for regularization (Zhang & Wallace, 2015). L2 norm[6] is calculated as the square root of the sum of the square vector values: $\|v2\| = sqrt(a_1^2,\ a_2^2,\ ...,a_n^2)$, where $\overrightarrow{v} = (a_1,\ a_2,\ ...,\ a_n)$.

We use the grid search capability to tune the hyperparameters of deep learning model. Fig. 4 shows a sample of a hyperparameter optimization procedure.

4.3.1. Model Variations: effect of word-embedding, word and sentence levels features

To analyze the effect of the word embedding feature (WEF), sentence-level features (SLF), Word-level features (WLF) we train different RNSA methods with different combinations of features as follows:

[6] https://machinelearningmastery.com/vector-norms-machine-learning/.

```
mod = KerasClassifier (build_fn=deepNN, epochs=100, batch_size=10, verbose=1)
learn_rate = [0.001, 0.01, 0.02, 0.03,0.1, 0.2, 0.3]
momentum = [0.0, 0.2, 0.4, 0.6, 0.8, 0.9]
weight_constraint = [1, 2, 3, 4, 5]
dropout_rate = [0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
neurons = [10, 15, 20, 25, 30]
init_mode = ['uniform', 'lecun_uniform', 'normal', 'zero', 'glorot_normal']
activation = ['softmax', 'relu', 'tanh', 'sigmoid', 'hard_sigmoid']
param_grid=dict(learn_rate=learn_rate,momentum=momentum,init_mode=init_mode,
                activation=activation,dropout_rate=dropout_rate,
                weight_constraint=weight_constraint,neurons=neurons)
grid = GridSearchCV(estimator=mod, param_grid=param_grid, n_jobs=1)
grid_result = grid.fit(x_train, y_train)
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

Fig. 4. The hyperparameter optimization procedure.

- RNSA $_{\text{Full}}$: This method incorporates all the available features: WEF, SLF and WLF.
- RNSA $_{\text{WEF+SLF}}$: This represents the RNSA method in the absence of the WLF feature set. It is trained and evaluated with the WEF and SLF features.
- RNSA $_{\text{WEF+WLF}}$: A RNSA method with WEF and WLF features.
- RNSA $_{\text{WEF}}$: A RNSA method that is trained and evaluated using WEF features.

The results of varying the RNSA method are presented in Table 3. As shown in Table, among all the various RNSA methods, the RNSA $_{Full}$ (*F_measure:* 0.7478) achieves the best performance in comparison with the other methods. It outperforms RNSA $_{WEF}$ by 6.71 points, RNSA $_{WEF+WLF}$ by 2.32 points and reports approximately 4.3 points better accuracy as compared to RNSA $_{WEF+SLF}$. However, (a) we get the best accuracy when we use all the features. This can be explained by the fact that RNSA Full exploits contextual information, sentiment information of sentences and words. It also is able to distinguish words with opposite sentiment polarity. (b) Word-level features have performed significantly better than sentence-level features. (c) Due to the results, we use the combined features (*WEF + WLF + SLF*) for the proposed method.

4.3.2. Comparison with related methods

In this subsection, we compare our method, RNSA, with the other existing well-known methods for sentence-level sentiment classification: (1) CNNSC (Kim, 2014); (2) IWVSA (Rezaeinia et al., 2017); (3) DLUSA (Di Capua & Petrosino, 2016); (4) CSNSA (Teng, Vo, & Zhang, 2016); (5) TSND (Li, Luong, Jurafsky, & Hovy, 2015); (6) CCRSA (Wang, Jiang, & Luo, 2016). Table 4 shows the performance of the RNSA against other methods in terms of precision, recall and *F*_measure. In Table 4, the column ("RNSA *improvement*") indicates the relative improvement for comparison between the RNSA method and other methods. The relative improvement measure is computed as follows: $(\frac{\text{Our Method} - \text{Other method}}{\text{Other method}}) \times 100$. The sign '+' demonstrates the RNSA improves the corresponding method. For instance, the RNSA improves the performance of the CCRSA method as follows: 7.02% (*F*_measure).

4.3.3. Discussion

From Table 4, we obtained the following observations. The RNSA outperforms other methods and obtained good performance. This is due to the facts that,

(1) Since (a) word2vec method does not explicitly exploit the sentiment information of the text, (b) a vector representation based on the word2vec method has not enough sentiment information to perform sentiment analysis and, (c) in vector representation using word2vec method, the words with opposite polarity such as '*good*' and '*bad*' are mapped to close word vectors, the RNSA method incorporates the prior sentiment knowledge information (*a sentiment lexicon*) into the word2vec method to cope the above-mentioned problems, while other methods (*e.g., TSND, CCRSA, CNNSC, DLUSA*) do not use the sentiment knowledge information.
(2) The RNSA method also combines ten sentiment dictionaries in order to use as the prior sentiment knowledge. The underlying

Table 3
The results obtained from the experiments carried out varying the RNSA method.

Methods	WEF	WLF	SLF	Precision	Recall	*F*_measure
RNSA $_{Full}$	+	+	+	0.8675	0.6571	0.7478
RNSA $_{WEF+SLF}$	+		+	0.8176	0.6193	0.7048
RNSA $_{WEF+WLF}$	+	+		0.8404	0.6369	0.7246
RNSA $_{WEF}$	+			0.7869	0.5998	0.6807

Table 4
The comparisons between our method (RNSA) and other methods.

Group	Methods	Precision	Recall	*F*_measure	RNSA improvement (%)
CNN	CNNSC *(CNN)*	0.7826	0.6004	0.6795	+ 10.05
	IWVSA *(CNN)*	0.7995	0.6132	0.6941	+ 7.74
DBN	DLUSA *(DBN)*	0.7469	0.5733	0.6487	+ 15.28
Recurrent	RNSA *(LSTM)*	0.8675	0.6571	0.7478	–
	CSNSA *(bi-LSTM)*	0.8205	0.6212	0.7071	+ 5.76
Recursive	TSND *(Tree bi-LSTM)*	0.7915	0.6071	0.6871	+ 8.82
CNN-RNN	CCRSA *(CNN-LSTM)*	0.8064	0.6164	0.6987	+ 7.02

reason is that, (a) to overcome the word coverage limit, (b) various sentiment dictionaries complement each other, while other methods (*TSND, CCRSA, CNNSC, DLUSA*) do not use any sentiment lexicon, CSNSA (*4 sentiment dictionaries*) and IWVSA (*6 sentiment dictionaries*) use for sentiment analysis.

(3) In addition, since the word embedding model cannot differentiate the senses of a word and creates a single representation per word form, the RNSA method can effectively deal with word sense variations, while the other methods (*e.g., CSNSA, TSND, CCRSA, CNNSC, DLUSA*) do not consider the aforementioned issue.

(4) Furthermore, since the performance of a sentiment analysis method relies on the different types of sentences, the RNSA method considers the types of sentences in sentiment analysis, while other methods do not consider.

(5) The RNSA also considers contextual polarity and the sentiment shifter rules in sentiment analysis, while other methods (*e.g., CSNSA, TSND, CCRSA: excluding the sentiment shifter rules*) and (*e.g., DLUSA, CNNSC, IWVSA: excluding the sentiment shifter rules and contextual polarity*) do not consider in sentiment analysis.

(6) RNN can handle sequential data while CNN cannot. Therefore, the RNSA method can capture word order information and semantic relationships between words which are very important for text classification, while other method (*e.g., IWVSA, CNNSC, DLUSA*) cannot.

(7) Since word2vec method are good at capturing the semantic information, the addition of handcrafted features assists it in finding the sentiment more accurately (Table 3). The RNSA method also uses complex external features (*word level and sentence levels features*), while other methods don't use external features.

4.3.4. Sentiment lexicon size

We also investigated whether bigger sentiment lexicon can lead to promising results. The relationship between the number of lexicon words and the performance is presented in Fig. 5. We see that the smallest dictionaries (AFINN (*word size:* 2477)) perform poorly. There seems to be a correlation that larger is better. It can be observed that the SentiWordNet (*word size:* 155,287) leads to the highest throughput. The reason is due to the wide coverage of sentiment words.

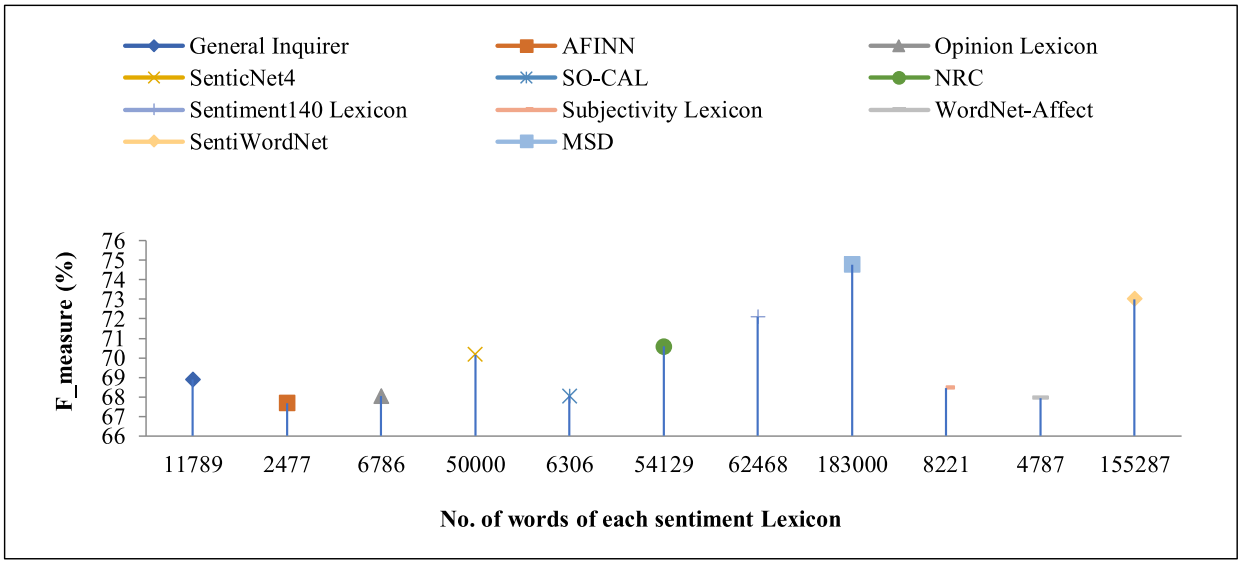

Fig. 5. Performance and the size of sentiment dictionary words.

4.4. RNSA implementation

In this research, we consider the following hypotheses: (1) a unified feature set which is constructed using the sentiment-based, word embedding-based, statistical and linguistic knowledge-based feature vectors, can improve the performance of a multi-documents opinion-oriented summarization model. (2) The integration of multiple strategies (*e.g., sentiment shifter rules, words with similar semantic context but opposite sentiment polarity, contextual polarity, sentence types, word sense variations, word order information and semantic relationships between words*) aids the proposed method to achieve superior performance. (3) A greater sentiment lexicon can generate a significant outcome. The current hypotheses need to be proven by an effective classification method and testing environment. By taking this goal, the RNSA method is proposed in this research. The proposed method is further tested over Movie Review Dataset.

Finally, we obtained the following significant implications of the current research: (1) The greater sentiment lexicon can result a promising outcome; (2) the integration of the multiple strategies as mentioned above has a promising impact on sentiment analysis; (3) the effect of the integration of several valuable resource-information such as word embedding, sentiment information, statistical and linguistic information on opinion-oriented summarization method; (4) the experiments show very promising results closely comparable (better in some cases) to existing methods. Thus, it is necessary to consider multiple strategies as discussed above (hypothesis (2)) and employ several valuable resource-information (hypothesis (1)) in order to create a hybrid vector and achieve a remarkable performance. On the other hand, the method also needs a sentiment lexicon to compute sentence sentiment score and encode the sentiment information into the word embedding. Therefore, a greater sentiment lexicon can improve the method performance.

5. Conclusion and future work

The deep learning-based method has attracted lots of attentions recently as a hot research topic in many fields such as image processing, speech recognition and natural language processing. In this paper, we present a novel deep-learning-based method for sentiment classification. The RNSA includes two main parts: (1) *sentiment analysis*: to extract relevant features in order to determine sentence polarity. (2) *pre-processing*: it includes the basic linguistic functions. The RNSA incorporates various types of features: sentiment, statistical, linguistic knowledge and word-level embedding to create a hybrid vector representation of each sentence. It also combines several sentiment dictionaries to tackle the word coverage limit. Furthermore, the RNSA considers several strategies to cope the following problems: (a) contextual polarity; (b) sentiment shifter; (c) type of sentence; (d) word sense variations; (e) integration of sentiment information and word embeddings; (f) word order information and semantic relationships between words.

We conduct experiments on three open benchmark datasets: Movie dataset. The experimental results on the above-mentioned dataset assess and confirm the effectiveness of the RNSA. The results also show that the method is competitive with state-of-the-art to previous reported results. We proceeded with a comparison between different RNSA methods as follows: RNSA $_{\mathrm{WLF+SLF+WEF}}$, RNSA $_{\mathrm{WEF+SLF}}$, RNSA $_{\mathrm{WEF+WLF}}$, RNSA $_{\mathrm{WEF}}$, where word embedding feature (WEF), sentence-level features (SLF), Word-level features (WLF). The results show among all the various RNSA methods, the RNSA $_{\mathit{Full:}\ \mathrm{WLF+SLF+WEF}}$ achieves the best performance in comparison with the other methods.

In future work, although the RNSA has performed better than existing methods, we propose the following improvements to the method to enhance the performance of the RNSA. (a) We aim to investigate in more depth the problem of comparative and sarcastic sentence handling. (b) To consider a rich set of new features in order to improve the RNSA performance. (c) To measure the performance of the RNSA using large and various datasets.

Acknowledgement

This work is supported by The Ministry of Higher Education (MOHE) under Q.J130000.21A2.03E53 - STATISTICAL MACHINE LEARNING METHODS TO TEXT SUMMARIZATIONS and 13H82 (INTELLIGENT PREDICTIVE ANALYTICS FOR RETAIL INDUSTRY). The authors would like to thank Research Management Centre (RMC), Universiti Teknologi Malaysia (UTM) for the support in R & D, UTM Big Data Centre (BDC) for the inspiration in making this study a success. The authors would also like to thank the anonymous reviewers who have contributed enormously to this work.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.ipm.2019.02.018.

References

Abdi, A., Shamsuddin, S. M., & Aliguliyev, R. M. (2018a). QMOS: Query-based multi-documents opinion-oriented summarization. *Information Processing & Management, 54*(2), 318–338.

Abdi, A., Shamsuddin, S. M., Hasan, S., & Piran, J. (2018). Automatic sentiment-oriented summarization of multi-documents using soft computing. *Soft Computing, 4*, 1–18.

Abdi, A., Shamsuddin, Siti M., Hasan, S., & Piran, J. (2018b). Machine learning-based multi-documents sentiment-oriented summarization using linguistic treatment. *Expert Systems with Applications, 109*, 66–85.

Ain, Q. T., Ali, M., Riaz, A., Noureen, A., Kamran, M., Hayat, B., et al. (2017). Sentiment analysis using deep learning techniques: A review. *International Journal of*

Advanced Computer Science and Applications, 8(6), 424.
Al-Sallab, A., Baly, R., Hajj, H., Shaban, K. B., El-Hajj, W., & Badaro, G. (2017). AROMA: a recursive deep learning model for opinion mining in Arabic as a low resource language. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP), 16*(4), 25.
Ahmad, A. S., Hajj, H., Badaro, G., Baly, R., El Hajj, W., & Shaban, K. B. (2015). Deep learning models for sentiment analysis in Arabic. *Proceedings of the second workshop on Arabic natural language processing*.
Amir, S., Almeida, M. B., Martins, B., Filgueiras, J., & Silva, M. J. (2014). TUGAS: exploiting unlabelled data for Twitter sentiment analysis. *Proceedings of the SemEval@ COLING*.
Araque, O., Corcuera-Platas, I., Sanchez-Rada, J. F., & Iglesias, C. A. (2017). Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications, 77*, 236–246.
Baccianella, S., Esuli, A., & Sebastiani, F. (2010). SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. *Proceedings of the LREC*.
Bansal, M., Gimpel, K., & Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. *Proceedings of the fifty-second annual meeting of the association for computational linguistics (volume 2: short papers)*.
Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks, 5*(2), 157–166.
Bermingham, A., & Smeaton, A. F. (2010). Classifying sentiment in microblogs: Is brevity an advantage? *Proceedings of the nineteenth ACM international conference on information and knowledge management*.
Bishop, C., & Bishop, C. M. (1995). *Neural networks for pattern recognition.* Oxford University Press.
Caliskan, A., Bryson, J. J., & Narayanan, A. (2017). Semantics derived automatically from language corpora contain human-like biases. *Science, 356*(6334), 183–186.
Cambria, E., Poria, S., Bajpai, R., & Schuller, Björn W. (2016). SenticNet 4: a semantic resource for sentiment analysis based on conceptual primitives. *Proceedings of the COLING*.
Chen, T., Xu, R., He, Y., & Wang, X. (2017). Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN. *Expert Systems with Applications, 72*, 221–230.
Chen, T., Xu, R., He, Y., Xia, Y., & Wang, X. (2016). Learning user and product distributed representations using a sequence model for sentiment analysis. *IEEE Computational Intelligence Magazine, 11*(3), 34–44.
Cliche, M. (2017). BB_twtr at SemEval-2017 task 4: Twitter sentiment analysis with CNNs and LSTMs. *Proceedings of the 11th international workshop on semantic evaluations (SemEval-2017)* (pp. 573–580). .
Day, M.-Y., & Lin, Y.-Da (2017). Deep learning for sentiment analysis on Google play consumer review. *Proceedings of the 2017 IEEE international conference on information reuse and integration (IRI)*.
Deshwal, A., & Sharma, S. K. (2016). Twitter sentiment analysis using various classification algorithms. *Proceedings of the fifth international conference on reliability, Infocom technologies and optimization (trends and future directions) (ICRITO)*.
Di Capua, M., & Petrosino, A. (2016). A deep learning approach to deal with data uncertainty in sentiment analysis. *Proceedings of the international workshop on fuzzy logic and applications*.
Du, X., Cai, Y., Wang, S., & Zhang, L. (2016). Overview of deep learning. *Proceedings of the youth academic annual conference of Chinese association of automation (YAC)*.
Giatsoglou, M., Vozalis, M. G., Diamantaras, K., Vakali, A., Sarigiannidis, G., & Chatzisavvas, K. C. (2017). Sentiment analysis leveraging emotions and word embeddings. *Expert Systems with Applications, 69*, 214–224.
Gupta, P., Tiwari, R., & Robert, N. (2016). Sentiment analysis and text summarization of online reviews: A survey. *Proceedings of the 2016 international conference on communication and signal processing (ICCSP)*.
Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580.
Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735–1780.
Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. *Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining*.
Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. *Proceedings of the European conference on machine learning*.
Kamkarhaghighi, M., & Makrehchi, M. (2017). Content tree word embedding for document representation. *Expert Systems with Applications, 90*, 241–249.
Khan, F. H., Qamar, U., & Bashir, S. (2016). A semi-supervised approach to sentiment analysis using revised sentiment strength based on SentiWordNet. *Knowledge and Information Systems, 51*(3), 1–22.
Kim, Y. (2014). Convolutional neural networks for sentence classification. *proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (pp. 1746–1751). .
Li, J., Luong, M.-T., Jurafsky, D., & Hovy, E. (2015). When are tree structures necessary for deep learning of representations? *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2304–2314). .
Lin, C.-C., Ammar, W., Dyer, C., & Levin, L. (2015). Unsupervised pos induction with word embeddings. *Proceedings of the 2015 annual conference of the North American chapter of the ACL* (pp. 1311–1316). .
Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies, 5*(1), 1–167.
Liu, Y., Liu, B., Shan, L., & Wang, X. (2018). Modelling context with neural networks for recommending idioms in essay writing. *Neurocomputing, 275*, 2287–2293.
Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. *Proceedings of the forty-ninth annual meeting of the association for computational linguistics: Human language technologies-volume 1*.
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval, 1*. Cambridge: Cambridge University Press.
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. *Proceedings of the advances in neural information processing systems*.
Miura, Y., Sakaki, S., Hattori, K., & Ohkuma, T. (2014). TeamX: A sentiment analyzer with enhanced lexicon mapping and weighting scheme for unbalanced data. *Proceedings of the SemEval@ COLING*.
Mohammad, S. M., Kiritchenko, S., & Zhu, X. (2013). NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. arXiv preprint arXiv:1308.6242.
Narayanan, R., Liu, B., & Choudhary, A. (2009). Sentiment analysis of conditional sentences. *Proceedings of the 2009 conference on empirical methods in natural language processing. 1*.
Nguyen, H., & Nguyen, M.-L. (2017). A deep neural architecture for sentence-level sentiment classification in Twitter social networking. *Proceedings of the international conference of the Pacific association for computational linguistics*.
Nielsen, F. Å., (2011). A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. arXiv preprint arXiv:1103.2903.
Rana, T. A., & Cheah, Y.-N. (2016). Aspect extraction in sentiment analysis: comparative analysis and survey. *Artificial Intelligence Review, 46*(4), 459–483.
Rezaeinia, S. M., Ghodsi, A., & Rahmani, R. (2017). Improving the accuracy of pre-trained word embeddings for sentiment analysis. arXiv preprint arXiv:1711.08609.
Riloff, E., & Wiebe, J. (2003). Learning extraction patterns for subjective expressions. *Proceedings of the 2003 conference on empirical methods in natural language processing*.
Rojas-Barahona, L. M. (2016). Deep learning for sentiment analysis. *Language and Linguistics Compass, 10*(12), 701–719.
Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks, 61*, 85–117.
Stone, P. J., & Hunt, E. B. (1963). A computer approach to content analysis: Studies using the general inquirer system. *Proceedings of the May 21–23, 1963, spring joint computer conference*.
Strapparava, C., & Valitutti, A. (2004). WordNet affect: an affective extension of wordnet. *Proceedings of the LREC*.
Sun, S., Luo, C., & Chen, J. (2017). A review of natural language processing techniques for opinion mining systems. *Information Fusion, 36*, 10–25.
Sun, X., Li, C., & Ren, F. (2016). Sentiment analysis for Chinese microblog based on deep neural networks with convolutional extension features. *Neurocomputing, 210*,

227–236.
Taboada, M., Brooke, J., Tofiloski, M., Voll, K., & Stede, M. (2011). Lexicon-based methods for sentiment analysis. *Computational Linguistics, 37*(2), 267–307.
Teng, Z., Vo, D. T., & Zhang, Y. (2016). Context-sensitive lexicon features for neural sentiment analysis. *Proceedings of the 2016 conference on empirical methods in natural language processing*.
Tripathy, A., Agrawal, A., & Rath, S. K. (2016). Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications, 57*, 117–126.
Vateekul, P., & Koomsubha, T. (2016). A study of sentiment analysis using deep learning techniques on Thai Twitter data. *Proceedings of the thirteenth international joint conference on computer science and software engineering (JCSSE)*.
Wang, X., Jiang, W., & Luo, Z. (2016). Combination of convolutional and recurrent neural network for sentiment analysis of short texts. *Proceedings of COLING 2016, the twenty-sixth international conference on computational linguistics: technical papers*.
Xia, R., Xu, F., Yu, J., Qi, Y., & Cambria, E. (2016). Polarity shift detection, elimination and ensemble: A three-stage model for document-level sentiment analysis. *Information Processing & Management, 52*(1), 36–45.
Xiao, Z., Li, X., Wang, L., Yang, Q., Du, J., & Sangaiah, A. K. (2018). Using convolution control block for Chinese sentiment analysis. *Journal of Parallel and Distributed Computing, 116*, 18–26.
Xie, Y., Chen, Z., Zhang, K., Cheng, Y., Honbo, D., Agrawal, A., et al. (2014). MuSES: Multilingual sentiment elicitation system for social media data. *IEEE Intelligent Systems, 29*(4), 34–42.
Yadav, N., & Chatterjee, N. (2016). Text summarization using sentiment analysis for DUC data. *Proceedings of the 2016 international conference on information technology (ICIT)*.
Zhang, Q., Yang, L. T., Chen, Z., & Li, P. (2018). A survey on deep learning for big data. *Information Fusion, 42*, 146–157.
Zhang, Y., & Wallace, B. (2015). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. arXiv preprint arXiv:1510.03820.