IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 3, 2019, accepted December 26, 2019, date of publication January 3, 2020, date of current version January 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2019.2963702

Improving the Performance of Sentiment Analysis of Tweets Containing Fuzzy Sentiment Using the Feature Ensemble Model

HUYEN TRANG PHAN[1], VAN CUONG TRAN[2], NGOC THANH NGUYEN[3,4], (Senior Member, IEEE), AND DOSAM HWANG[1]

[1]Department of Computer Engineering, Yeungnam University, Gyeongsan 38541, South Korea
[2]Faculty of Engineering and Information Technology, Quang Binh University, Dong Hoi 47000, Vietnam
[3]Faculty of Computer Science and Management, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland
[4]Faculty of Information Technology, Nguyen Tat Thanh University, Ho Chi Minh 70000, Vietnam

Corresponding author: Dosam Hwang (dshwang@yu.ac.kr)

This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT and Future Planning under Grant 2017R1A2B4009410.

ABSTRACT The increase in the volume of user-generated content on Twitter has resulted in tweet sentiment analysis becoming an essential tool for the extraction of information about Twitter users' emotional state. Consequently, there has been a rapid growth of tweet sentiment analysis in the area of natural language processing. Tweet sentiment analysis is increasingly applied in many areas, such as decision support systems and recommendation systems. Therefore, improving the accuracy of tweet sentiment analysis has become practical and an area of interest for many researchers. Many approaches have tried to improve the performance of tweet sentiment analysis methods by using the feature ensemble method. However, most of the previous methods attempted to model the syntactic information of words without considering the sentiment context of these words. Besides, the positioning of words and the impact of phrases containing fuzzy sentiment have not been mentioned in many studies. This study proposed a new approach based on a feature ensemble model related to tweets containing fuzzy sentiment by taking into account elements such as lexical, word-type, semantic, position, and sentiment polarity of words. The proposed method has been experimented on with real data, and the result proves effective in improving the performance of tweet sentiment analysis in terms of the F_1 score.

INDEX TERMS Feature ensemble model, fuzzy sentiment, tweet embeddings, tweet sentiment analysis.

I. INTRODUCTION

With the growth of social networks, an increasing number of people want to find, share, and exchange information with each other without any regard to the geographical distance. Therefore, people need quick, free, and readily available tools to help them achieve these needs. Social networks can respond to these requirements of users. The number of users on social networks increases every day, and they tend to post every information about topics which they concern. This information is a significant source of data for people such as researchers, manufacturers, politicians, and celebrities. Currently, one of the most popular social network sites is Twitter [6]. Twitter's user activity has grown quickly, with approximately 500 million tweets published daily in 2014, the last time official stats were released.[1,2] According to statistics on April 17th, 2019,[3] the number of active Twitter users per month is 330 million worldwide for Q1 2019 (from January 1st to March 31th) versus 326 million for Q3 2018.

The associate editor coordinating the review of this manuscript and approving it for publication was Jon Atli Benediktsson.

[1]http://www.internetlivestats.com/twitter-statistics/
[2]https://www.businessofapps.com/data/twitter-statistics/
[3]https://zephoria.com/twitter-statistics-top-ten/

The information on Twitter is a source that could provide many benefits if we know how to exploit it. Tweet sentiment analysis (TSA) is a research area that aims to analyze users' sentiment or opinions toward entities-such as topics, events, individuals, issues, services, products, and organizations-and their attributes based on the content of their tweets [3]. For the past few years, the prosperity of Twitter has propelled the development of TSA. This analysis can provide online advice and recommendations for both customers and merchants. For producers, sentiment analysis can be used to analyze their products and services based on e-commerce platforms on Twitter. Due to the virtual nature of online shopping, users are not easily able to determine whether a product is of good quality. Sentiment analysis can help users learn about the comments or opinions of other consumers.

A feature ensemble model is a combination of a set of models (base classifiers) to obtain a more accurate and reliable model in comparison with what a single model can achieve [2]. This model is used as a support tool for other models (especially as deep learning models) to solve many real tasks. Previous feature ensemble models employed in TSA mainly focused on extracting features from the text. Recently, word embeddings have been utilized as an alternative to the manual techniques [23], [28], [36]. Although the previous embedding pre-trained models, such as Word2Vec and GloVe, are very active, these methods have some limitations. The Word2Vec and GloVe models need massive data for training and creating a suitable vector for each word [1], [13]. Therefore, these methods may not be the best conform for small and informal data such as tweets. Besides, Word2Vec and GloVe ignore the context of the text [14]. Another problem is that both models do not consider the relationships between words that do not co-occur [8]. In addition, according to Araque *et al.* [1]; Giatsoglou *et al.* [13]; Ren *et al.* [27], a significant limitation with the Word2Vec and GloVe models is not identifying the sentiment information of the given text. Furthermore, according to Tang *et al.* [33], this omission is the cause that those words with inverse sentiment are converted into close vectors, which leads to the performance of sentiment analysis not high. Therefore, improving the word embedding techniques by considering the impact of the sentiments, the POS tags of the words, etc. is necessary.

Many approaches have been tested to improve the accuracy of TSA methods with relatively good results by using the feature ensemble method. However, most of these methods attempted to model the syntactic information of words while ignoring the sentiment context. In other words, there are some researchers who have been tried to build a feature ensemble model, but they have not fully considered the features, such as the lexical, word-type, semantic, position of words. Additionally, they have not yet mentioned the impact of the fuzzy sentiment phrases. This motivated us to propose a new approach to solve this problem. The contributions of this study can be summarized as follows. First, we built the feature ensemble model to translate each tweet into a vector (called tweet embeddings) by extracting the features related to tweets containing fuzzy sentiment, such as: 1) Part-of-Speech (POS) tags; 2) N-grams of words; 3) sentiment score of words such as negation words, fundamental sentiment words, and fuzzy semantic words; 4) the distance between words; and 5) words embeddings using the GloVe model. Creating tweet embeddings was the main contribution of our proposal. Then, the convolutional neural network (CNN) model with the input layer as tweet embeddings was used to improve the performance of sentiment analysis. This study was proposed based on the combination of the feature ensemble model, deep learning algorithm, and the divide-and-conquer strategy. The divide-and-conquer approach means that this study only concentrates on improving the performance of the sentiment analysis method applying to a specific type of tweet, i.e., tweets contain fuzzy sentiment phrases.

The rest of the paper is organized as follows. In Section 2, we summarize the literature related to sentiment analysis approaches. The research problem is described in Section 3 and the proposed method is presented in Section 4. The experimental results and evaluations are shown in Section 5. The conclusions and future work are discussed in the last section.

II. RELATED WORKS

In this section, we discuss some academic works that were the motivation for our proposal. We focus on analyzing the methods published to improve the performance of sentiment analysis based on the feature ensemble model and the divide-and-conquer strategy.

In order to improve the performance of the existing models, the combination models have been written about extensively in sentiment analysis such as in [2], [19], [23], [28], [29], [36]. Rehman *et al.* [28] provided a hybrid model using LSTM and a deep CNN model named Hybrid CNN-LSTM model to improve the accuracy of the sentiment analysis problem by using the word to vector approach to train first-word embeddings. Word embedding is combined with a set of features that are extracted by convolution and global max-pooling layers with long-term dependencies. The results show that this model outperforms traditional deep learning and machine learning techniques. Meanwhile, Ye *et al.* [36] proposed to combine sentiment information from the training data and a sentiment lexicon; this information was then encoded into word embeddings. This paper did not consider the effect of syntactic and semantic of words when extracting features. Jianqiang *et al.* [23] constructed a feature ensemble model by combining the word embeddings collected from the GloVe model with N-gram features and sentiment scores. The model achieved good results. However, the authors did not mention how the Twitter corpus was collected, and if the tweets contained sentiment or not. No experiments were conducted on the combination of the GloVe word embeddings with the manually extracted features and comparisons with previous works on the same datasets. Meanwhile, Hassan *et al.* [19] transformed words into real valued feature vectors that

capture semantic and syntactic information. However, this method only focused on the surface features of the word without considering the impact of the in-depth features. Hence, Al-Twairesh *et al.* [2] proposed a feature ensemble model by considering the surface and in-depth features. The surface features are manually extracted features, and the in-depth features are generic word embeddings and sentiment specific word embeddings. Rezaeinia *et al.* [29] proposed a novel method to increase the accuracy of pre-trained word embeddings in sentiment analysis based on POS tags of words, lexicon-based words, position-based words, and word embeddings. The authors tested the performance of the proposal with deep learning models. These approaches achieved state-of-the-art results on several benchmarking datasets for sentiment analysis. However, the authors analyzed the sentiment of general tweets without focusing on any specific kind of tweets. It is difficult to have a ''one-technique fits all'' approach because different types of sentences express sentiments in different ways. Thus, a divide-and-conquer approach is preferable [22], i.e., a study focusing on each type of sentence separately could perform sentiment analysis more accurately. To understand the strategy clearly, some related papers are analyzed as follows.

Some research focused on analyzing the sentiment analysis by applying the divide-and-conquer strategy, such as [10], [11], [22], [25], [26]. Narayanan *et al.* [22] presented a linguistic analysis of conditional sentences, and then built supervised learning models to determine if sentiments expressed on different topics are positive, negative, or neutral. Experimental results on conditional sentences from five diverse domains are conducted to demonstrate the effectiveness of the proposed approach. Ganapathibhotla and Liu [11] focused on determining which entities in comparison are preferred by users. The experiments using comparative sentences from product reviews and forum posts show that the approach is effective. Farías *et al.* [10] described a system for sentiment analysis of the figurative language used on Twitter at SemEval 2015. A distinctive feature of their approach is that they used sentiment word lexicons providing polarity annotations as well as newer sources for dealing with emotions and psycholinguistic information. The system also exploited novel and standard structural features of tweets. This paper obtained significant results in both ironic and sarcastic tweets. Phan *et al.* [25] tried to use advanced algorithms such as MLP and CNN to detect and analyze the sentiment of tweets containing conditional sentences. In the paper [26], a method to analyze the sentiment of tweets containing fuzzy sentiment phrases was utilized by calculating the score of fuzzy sentiment phrases.

As analyzed in the above literature, we can see that many studies improved the performance of sentiment analysis by using the feature ensemble model. Several methods obtained results of sentiment analysis based on the divide-and-conquer strategy, meaning each study focused on specific data. However, no study has tried to combine the feature ensemble model and the divide-and-conquer strategy for sentiment analysis. This motivated us to conduct research on this topic.

III. RESEARCH PROBLEMS

A. FUZZY SENTIMENT PHRASES AND RELATED LEXICONS

A fuzzy sentiment is a user's attitude toward something, but the attitude is not clearly expressed. It is usually represented by one or more fuzzy sentiment phrases.

Fuzzy sentiment phrases do not usually express emotion clearly. They comprise more than one word with at least one being a fundamental sentiment word and the remaining word(s) can be either a fuzzy semantic word or a combination of negation words and fuzzy semantic words [26]. Fuzzy sentiment phrases are divided into two main types as follows.

1) Fuzzy sentiment phrases are created by one fuzzy semantic word and one fundamental word as in the following examples: a) Intensifier word plus negative word, e.g., ''too bad''; b) Intensifier word plus positive word, e.g., ''so good''; c) Diminisher word plus negative word, e.g., ''fairly bad''; d) Diminisher word plus positive word, e.g., ''slightly good.'' 2) Fuzzy sentiment phrases are generated by one negation word, one fuzzy semantic word, and one fundamental word as in the following examples: a) Negation word plus intensifier word and negative word, e.g., ''not too bad''; b) Negation word plus diminisher word and plus negative word, e.g., ''not fairly bad''; c) Negation word plus intensifier word and plus positive word, e.g., ''not so good.'' d) Negation word plus diminisher word and plus positive word, e.g., ''not slightly good.'' The type of lexicons regarding fuzzy sentiment phrases were collected from different sources, in which the fundamental sentiment words were selected from SentiWordNet (SWN). SWN was proposed by Baccianella *et al.* [4] with more than 60,000 synsets and used in many research related to sentiment analysis of online reviews, such as in the papers [3], [5], and [18]. The fuzzy semantic words were created by combining the extracted words from three research in papers [3], [15], [32].

Fundamental sentiment words include positive words and negative words. Words such as ''angry,'' ''sad,'' and ''happy'' are used to express emotional states of users. Positive words have a positive sentiment attached to them. Similarly, negative words have a negative sentiment attached to them.

Negation words are words that stand before the fundamental sentiment words and change the polarity of these words, e.g., ''not'' and ''n't.''

Fuzzy semantic words are a set of words which increase or decrease the degree of the sentiment of fundamental sentiment words. This lexicon consists of the following two types of words: 1) Intensifier words are words standing before the fundamental sentiment words and can increase the polarity of these words, e.g., ''too,'' ''so,'' and ''overly''. 2) Diminisher words are words standing before the fundamental sentiment words and can decrease the polarity of these words, e.g., ''quite,'' ''fairly,'' and ''slightly'' [35].

B. SENTIMENT SCORE OF A WORD

Given a set of tweets $\mathcal{T}$.

For $t \in \mathcal{T}$: let $\mathcal{W}$ be a set of words existing in t.

For $w \in \mathcal{W}$: let S be a set of synsets of word w.

For $sn \in \mathcal{S}$:

let $\mathcal{P}$ be a set of POS tags of words in $\mathcal{W}$,

let $\mathcal{P}s$ be the positivity score assigned by SWN to synset sn,

let $\mathcal{N}s$ be the negativity score assigned by SWN to synset sn.

In which, $\mathcal{P}s, \mathcal{N}s \in [0.0, 1.0]$ and $\mathcal{N}s + \mathcal{P}s \leq 1.0$.

For $p \in \mathcal{P}$:

let $\mathcal{S}p(w, p)$ be a positive score of word w has POS tag p corresponding to synsets,

let $\mathcal{S}n(w, p)$ be a negative score of word w has POS tag p corresponding to synsets.

The positive and negative score of word w is computed as follows:

$$\mathcal{S}p(w, p) = \frac{1}{m} \sum_{sn \in \mathcal{S}} \mathcal{P}s(sn) \tag{1}$$

$$\mathcal{S}n(w, p) = \frac{1}{m} \sum_{sn \in \mathcal{S}} \mathcal{N}s(sn) \tag{2}$$

where m represents the number of synsets of word w.

Let $\mathcal{F}$ be a set of fundamental sentiment words. Let $\mathcal{F}s$ be a set of fuzzy semantic words. Let $\mathcal{N}$ be a set of negation words.

For $f \in \mathcal{F}$ and $p \in \mathcal{P}$: let $\mathcal{S}c(f, p)$ be the sentiment score of word f corresponding to POS tag p and $\mathcal{S}c(f, p)$ is computed based on $\mathcal{S}p(f, p)$, $\mathcal{S}n(f, p)$ as follows:

$$\mathcal{S}c(f, p) = \mathcal{S}p(f, p) - \mathcal{S}n(f, p) \tag{3}$$

Next, the sentiment score of the fuzzy semantic words are determined as follows:

For $fs \in \mathcal{F}s$: let $\mathcal{S}c(fs)$ be a sentiment score of fs. Throughout the experiment and as analyzed at above, the sentiment score of fundamental words will be in the range $[-0.75, 0.75]$; therefore the value of fuzzy semantic words was chosen in the range $[-0.25,0.25]$. In this study, we used English modifier words offered by Strohm and Florian in the paper [31] as fuzzy semantic words. We used the numeric values offered by [3], [15], [32] to assign the score for intensifier and diminisher word lists. We then normalized numeric scores to each fuzzy semantic word to fit with our proposal by mapping from range $[-100\%,+100\%]$ to range $[-0.25,0.25]$. The score of the fuzzy semantic words is shown in TABLE 1 and TABLE 2.

TABLE 1. The score for some intensifier words.

Word	List 1	List 2	Normalized List	Score
really	+15%	+15%	+15%	0.04
very	+25%	+50%	+38%	0.09
extraordinarily	+50%	+75%	+63%	0.16
most		+90%	+95%	0.24
completely		+100%	+100%	0.25
totally		+70%	+70%	0.18
too		+45%	+45%	0.11
extremely		+80%	+80%	0.2
pretty	-10%	+20%	+5%	0.01

List 1 is the score extracted from paper [17], [35],
List 2 is the score extracted from paper [3].

TABLE 2. The score for some diminisher words.

Word	List 1	List 2	Normalized List	Score
slightly	-50%	-40%	-45%	-0.11
somewhat	-30%		-30%	-0.08
hardly		-70%	-70%	-0.18
less		-50%	-50%	-0.13
quite		-20%	-20%	-0.05
minor		-30%	-30%	-0.08
a few		-25%	-25%	-0.06
a little		-40%	-40%	-0.1
some		-25%	-25%	-0.06
a bit		-35%	-35%	-0.09
low		-20%	-20%	-0.05

List 1 is the score extracted from paper [17], [35],
List 2 is the score extracted from paper [3].

C. FORMAL MODEL FOR BUILDING A FEATURE ENSEMBLE MODEL

In this section, we formally define the problem of the feature ensemble model for tweets containing fuzzy sentiment. As a computational problem, the feature ensemble model for tweets containing fuzzy sentiment assumes that the input is a set of tweets containing fuzzy sentiment $\mathcal{T}$.

For $t \in \mathcal{T}$ and $w \in \mathcal{W}$: let $l2v(w)$, $sy2v(w)$, $se2v(w)$, $ps2v(w)$, and $pl2v(w)$ be lexical, word-type, semantic, position, and sentiment polarity vectors of word w, respectively.

Definition 1: The lexical vector of a word w, denoted by $l2v(w)$, is a k-dimensional vector, indicating the TF-IDF value of N-grams of word w. Let l_1, l_2, l_3 be vectors containing the TF-IDF values for 1-gram, 2-grams, 3-grams of a word w, respectively. The lexical vector is defined as

$$l2v(w) = \{(l_1, l_2, l_3) | l_1 \in \mathcal{R}^1 \wedge \mathcal{U}(w) = l_1, l_2 \in \mathcal{R}^h \wedge \mathcal{B}(w) = l_2, l_3 \in \mathcal{R}^q \wedge \mathcal{T}(w) = l_3\} \tag{4}$$

where $1 + h + q = k$, and $\mathcal{U}, \mathcal{B}, \mathcal{T}$ are mapping functions from a word to vectors containing TF-IDF values for 1-gram, 2-grams, and 3-grams of word w, respectively.

Definition 2: The word-type vector of a word w, denoted by $sy2v(w)$, is a k-dimensional vector used to supplement the POS tag information of a word w for the GloVe vector. The word-type vector is defined as

$$sy2v(w) = \{v_p | v_p \in \mathcal{R}^k \wedge \mathcal{P}(w) = v_p\} \tag{5}$$

where $\mathcal{P}(w)$ is a mapping function from a word w to vector v_p indicating the POS tag of this word. In this case, v_p is a one-hot encoding vector where all the elements of the vector are 0 except one, which has value as 1 corresponding to a POS tag of this word in the considered tweet.

Definition 3: The position vector of a word w, denoted by $ps2v(w)$, is a k-dimensional vector, in which the i-th dimension is a numerical measure indicating the relative distance between word w and word w_i in tweet t. The position vector

is defined as

$$ps2v(w) = \{(d_1, d_2, \ldots, d_n) | (d_1, d_2, \ldots, d_n) \in \mathcal{R}^{\mathcal{K}} \wedge \mathcal{D}(w, w_i) = d_i, i = 1, .., n\} \quad (6)$$

where w is a current word. $\mathcal{D}(w, w_i)$ is a function used to compute the distance from word w to word w_i, and

$$d_i = \frac{position(w) - position(w_i)}{Card(max_{t \in \mathcal{T}} length(t))} \quad (7)$$

with *position* is a function determining the order of word in a tweet. $Card(max_{t \in \mathcal{T}} length(t))$ is a function to give the number of words of the most length tweet.

Definition 4: The sentiment polarity vector of a word w, denoted by $pl2v(w)$, is a $\mathcal{K}$-dimensional vector, indicating the sentiment score of word w. The sentiment polarity vector is defined as

$$pl2v(w) = \{sc_w | sc_w \in \mathcal{R}^{\mathcal{K}} \wedge \mathcal{S}(w) = sc_w\} \quad (8)$$

where $\mathcal{S}(w)$ is a mapping function from word w to vector sc_w determining the sentiment score of this word and

$$sc_w = \begin{cases} \mathcal{S}c(w, p), & if \ w \in \mathcal{F}, \\ \mathcal{S}c(w), & if \ w \in \mathcal{F}s, \\ 1, & if \ w \in \mathcal{N}, \\ 0, & if \ w \notin \mathcal{N} \ and \ w \notin \mathcal{F} \cup \mathcal{F}s. \end{cases} \quad (9)$$

Definition 5: The semantic vector of a word w, denoted by $se2v(w)$, is a $\mathcal{K}$-dimensional vector, indicating GloVe word embeddings of word w. The semantic vector is defined as

$$se2v(w) = \{se_w | se_w \in \mathcal{R}^{\mathcal{K}} \wedge \mathcal{S}e(w) = se_w\} \quad (10)$$

where $\mathcal{S}e(w)$ is a mapping function from word w to vector se_w determining the context of this word and

$$se_w = \begin{cases} gloveVec(w), & if \ w \in GloVe, \\ randomVec(w), & if \ w \notin GloVe. \end{cases} \quad (11)$$

Why do we choose random vector without assigning a vector of zero values, or vector of particular numbers for unknown words? We briefly explain as follows: If we assign a vector of zero values or vector of very specific numbers for all unknown words, it will be the cause that the different words are mapped into close vectors, and the CNN model will understand that these words are the same word. Meanwhile, if we assign each vector of particular numbers corresponding to one unknown word, it will take too much time to search and assign word-by-word because there are quite many unknown words. In this case, we see that it will be best if we assign a random vector for unknown words.

Definition 6: A vector of a word w, denoted by $v(w)$, is a translation of word w into a d-dimensional vector by concatenating five feature vectors such as $l2v$, $sy2v$, $ps2v$, $pl2v$, $se2v$. The word vector is defined as

$$v(w) = \{v_w | v_w \in \mathcal{R}^d \wedge v_w = l2v(w) \oplus sy2v(w) \oplus ps2v(w) \oplus pl2v(w) \oplus se2v(w)\} \quad (12)$$

Definition 7: Tweet embedding of a tweet t, denoted by $\mathcal{T}2\mathcal{V}(t)$, is a translation of tweet into a vector by concatenating the word vectors $v(w_i)$, $i = 1, .., n$. Tweet embedding $\mathcal{T}2\mathcal{V}(t)$ is defined as

$$\mathcal{T}2\mathcal{V}(t) = \{tv_t | tv_t \in \mathcal{R}^{d \times n} \wedge tv_t = v(w_1) \oplus v(w_2) \oplus \ldots \oplus v(w_n)\} \quad (13)$$

where d is the dimension of $v(w_i)$, and n is the number of words.

D. RESEARCH QUESTION

To improve the accuracy of analyzing sentiment in tweets containing fuzzy sentiment of previous method, the main question for the research is as follows: *How can we improve the performance of analyzing the sentiment of tweets containing fuzzy sentiment based on the feature ensemble model?* This question is partitioned into the two following sub-questions:

The first question: *How can a feature ensemble model based on a set of features extracted from tweets be built?*

The second question: *How can the feature ensemble model be used to improve the accuracy of the sentiment analysis method applying to tweets containing fuzzy sentiment?*

IV. PROPOSED METHOD

In this section, we present a methodology to improve the accuracy of our previous proposal. The workflow of the method is shown in FIGURE 1. Our proposed method consists of three main steps: 1) a set of features related to tweets containing fuzzy sentiment are extracted; 2) a feature ensemble model to create tweet embeddings is proposed by combining feature vectors extracted in the first step; 3) a CNN model is used to classify the sentiment of tweets into five sets such as negative tweets set, neutral tweets set, positive tweets set, strong positive tweets set, and strong negative tweets set. The steps are detailed in the next sub-sections.

A. CREATING TWEET EMBEDDINGS

Tweet embeddings is the result of the feature ensemble model by concatenating five corresponding vectors such as $l2v$, $sy2v$, $pl2v$, $ps2v$, and $se2v$ into one vector.

1) LEXICAL VECTOR ($l2v$)

$l2v$ is built based on the extension of N-grams model called syntactic N-grams in paper [30]. The N-grams model is one of the most effective and straightforward representation models used in tweet sentiment analysis methods. In this study, N-grams, including 1-gram, 2-grams, and 3-grams, are used to map a word into a vector of the TF-IDF values of N-grams related to the word. For each word in a tweet, each N-gram related to this word becomes an entry in the feature vector with the corresponding feature value of TF-IDF.

2) WORD-TYPE VECTOR ($sy2v$)

$sy2v$ is built based on a POS tag of a word in tweet t. The POS tag is an essential and effective step in tweet sentiment

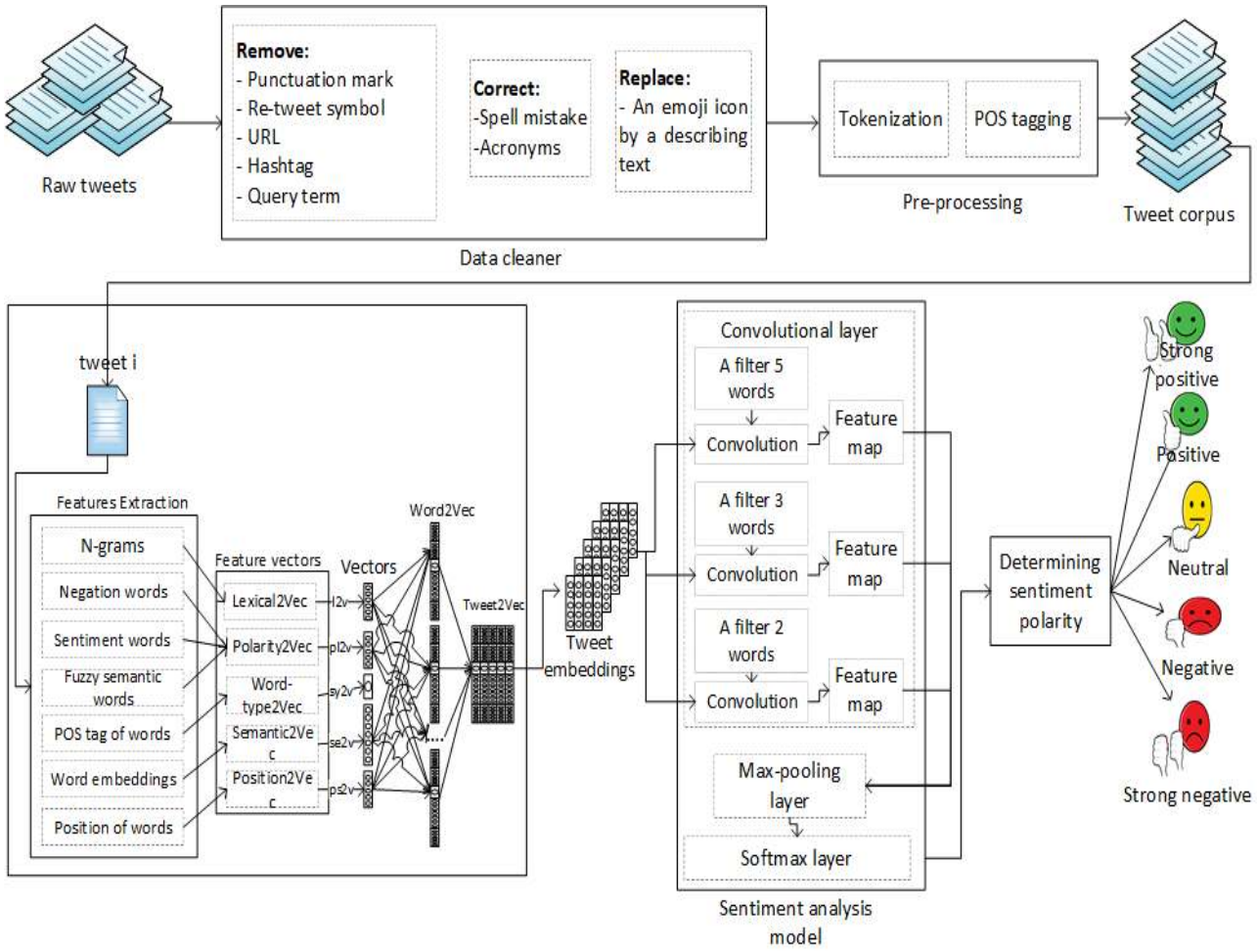

FIGURE 1. The workflow of proposed method.

TABLE 3. Example of the POS embedding Table.

	CD	DT	IN	JJ	TO	MD	NN	PRP	SYM	TO	VB	...	WRB
I	0	0	0	0	0	0	0	1	0	0	0	...	0
have	0	0	0	0	0	0	0	0	0	0	1	...	0
a	0	1	0	0	0	0	0	0	0	0	0	...	0
good	0	0	0	1	0	0	0	0	0	0	0	...	0
phone	0	0	0	0	0	0	1	0	0	0	0	...	0
.	0	0	0	0	0	0	0	0	1	0	0	...	0

analysis, which is the process of assigning each word according to a proper POS tag. The POS tag gives much information related to a word, such as its neighbors, syntactic categories (nouns, verbs, adjectives, adverbs, etc.), and similarities and dissimilarities between them. In addition, fundamental sentiment words may be used in multiple contexts, not all of which may correspond to an opinion. The NLTK toolkit [7] is used to annotate the POS tags. Each generated POS tag is then converted into a one hot vector. For example, assume that there is a tweet, ''I have a good phone.'' The word-type vector is determined based on the POS embedding table (TABLE 3) as follows:

From TABLE 3, $sy2v(good) = (0,0,0,1,0,0,0,0,0,0,0,\ldots,0)$

3) POLARITY SENTIMENT VECTOR ($pl2v$)

$pl2v$ is built by extracting features related to information such as negation words, fundamental sentiment words, and fuzzy semantic words.

Negation words are explained as follows. This feature is extracted by using a window of 3 to 5 words before a sentiment word and search forth is kind of words.

Fuzzy semantic words are explained as follows. This feature is extracted by using a window size of 1 to 3 words before a sentiment word and search for these kinds of words. The appearance of fuzzy semantic words in the tweet and their score become features and feature values, respectively.

Fundamental sentiment words are explained as follows. The fundamental sentiment words and their sentiment score are used as the feature and the feature value, respectively.

4) SEMANTIC VECTOR ($se2v$)

$se2v$ is built based on the word embeddings. The 300-dimensional pre-trained word embeddings from GloVe[4]

[4]http://nlp.stanford.edu/projects/glove/

are used to compute a word embedding. The GloVe model was proposed by Pennington *et al.* [24] and used in many research related to tweet sentiment analysis with quite good performance. The GloVe model is a global log bilinear regression model that combines the advantages of the two major model families in the literature: local context window and global matrix factorization methods. The model efficiently utilizes statistical information by training the non-zero elements in a word-word co-occurrence matrix only, rather than on the entire sparse matrix or individual context windows in a large corpus. The model determines the word vector with ratios of co-occurrence probability rather than the possibility itself.

5) POSITION VECTOR ($ps2v$)

$ps2v$ is built based on the word position. The position information of words is useful for convolutional encoders since they give a sense of the portion of the sequence in the input or output [12]. The position of a word is found based on the relative distances of this word to the remaining words in a tweet. For the tweet, "I have a good phone.", the position vector of the word is determined based on the position embedding table (TABLE 4) as follows:

TABLE 4. Example of the position embedding Table.

	I	have	a	good	phone	.
I	0	1	2	3	4	5
have	-1	0	1	2	3	4
a	-2	-1	0	1	2	3
good	-3	-2	-1	0	1	2
phone	-4	-3	-2	-1	0	1
.	-5	-4	-3	-2	-1	0

From TABLE 4 and equation 7, assume that $Card(max_{t\in\mathcal{T}} length(t)) = 10$, we have $ps2v(good) = (-0.3,-0.2,-0.1,0,0.1,0.2)$

Step by step to build the feature ensemble model is shown in Algorithm 1.

B. ANALYZING SENTIMENT OF TWEETS CONTAINING FUZZY SENTIMENT

The CNN model is used to analyze the sentiment of tweets containing fuzzy sentiment. This model has become a significant deep learning model used in the NLP field since the research by Mohammad *et al.* [21] and Kim [16], who applied the success of CNN in sentiment analysis [9], [34]. The sentiment analysis model is built as the following phases:

1) TWEET EMBEDDINGS LAYER

Each tweet will be represented by a vector $\mathcal{T2V}$ by concatenating five feature vectors including $l2v$, $sy2v$, $ps2v$, $pl2v$, and $se2v$. The vector $\mathcal{T2V}$ is presented as follows:

$$\mathcal{T2V}_{1:n} \in \mathcal{R}^{d\times n} \wedge \mathcal{T2V}_{1:n} = v_1 \oplus v_2 \oplus v_3 \oplus \ldots \oplus v_n \quad (14)$$

where $\oplus$ is the concatenation operator, d is the demension of v_i, $v_i = l2v(w) \oplus sy2v(w) \oplus pl2v(w) \oplus ps2v(w) \oplus se2v(w)$ $(v_i = v(w_i))$.

Algorithm 1 Creating Tweet Embeddings

Require:
$\mathcal{W} = \{w_1, w_2, \ldots, w_n\}$, a set of words in tweet t
$\mathcal{P} = \{p_1, p_2, \ldots, p_m\}$, a set of POS tags of words
$\mathcal{N} = \{n_1, n_2, \ldots, n_{fi}\}$, a set of negation words
$\mathcal{F} = \{f_1, f_2, \ldots, f_k\}$, a set of fundamental sentiment words
$\mathcal{Fs} = \{fs_1, fs_2, \ldots, fs_l\}$, a set of fuzzy semantic words
Ensure: $\mathcal{T2V}$: Tweet embeddings
for $i = 1$ to m **do**
  $v_i \leftarrow PosVector(p_i)$
  $p_i \leftarrow \langle p_i, v_i \rangle$
end for
for $i = 1$ to n **do**
  $g_i \leftarrow PsVector(ps_i)$
  $ps_i \leftarrow \langle ps_i, g_i \rangle$
end for
for $z = 1$ to n **do**
  $l_1 \leftarrow \mathcal{U}(w_z)$, a vector of 1-gram regarding word w_z
  $l_2 \leftarrow \mathcal{B}(w_z)$, a vector of 2-grams regarding word w_z
  $l_3 \leftarrow \mathcal{T}(w_z)$, a vector of 3-grams regarding word w_z
  insert l_1, l_2, l_3 into $l2v$
  $p \leftarrow extractPOS(w_z)$
  for $j = 1$ to m **do**
    if $p = p_j$ **then**
      insert v_j into $sy2v$
    end if
  end for
  $g \leftarrow extractPosition(w_z)$
  for $i = 1$ to n **do**
    if $g = ps_i$ **then**
      insert g_j into $ps2v$
    end if
  end for
  if $w_z \in \mathcal{F}$ or $w_z \in \mathcal{Fs}$ **then**
    $s \leftarrow extractScore(w_z)$
    insert s into $pl2v$
  else if $w_z \in \mathcal{N}$ **then**
    $s \leftarrow 1$
    insert s into $pl2v$
  else
    $s \leftarrow 0$
    insert s into $pl2v$
  end if
  if $w \in GloVe$ **then**
    $k \leftarrow gloveVec(w_z)$
    insert k into $se2v$
  else
    $k \leftarrow randomVec(w_z)$
    insert k into $se2v$
  end if
  insert $l2v$, $sy2v$, $ps2v$, $pl2v$, and $se2v$ into v_z
end for
for $z = 1$ to n **do**
  $\mathcal{T2V} \leftarrow v_1 \oplus v_2 \oplus \ldots \oplus v_n$
end for
return $\mathcal{T2V}$

2) CONVOLUTIONAL LAYER

This layer aims to create a feature map (c) from the tweet embedding layer. The feature map is created by using a window of length q words from i to $i+q-1$ to slide and filter important features. Each time sliding of the window creates a new feature vector as follows:

$$c_i = \mathcal{ReLU}(\mathcal{M}.\mathcal{T2V}_{i:i+q-1} + b) \tag{15}$$

where $\mathcal{ReLU}$ is a rectified linear function. b is a bias term. $\mathcal{M} \in \mathcal{R}^{h \times qd}$ is a transition matrix created for each filter, h is the number of hidden units in the convolutional layer. Therefore, when a tweet is slided completely, the features map is generated as follows:

$$c = [c_1, c_2, .., c_{d-q+1}], \quad c \in \mathcal{R}^{d-q+1} \tag{16}$$

3) MAX-POOLING LAYER

The primary function of the max-pooling layer is to reduce the dimension of the feature map by taking the maximum value $\hat{c} = max(c)$ as the feature corresponding to each filter. Assume that we use m filters, after this step the obtained new feature is $\hat{c} = [\hat{c_1}, \hat{c_2}, .., \hat{c_m}]$. Then this vector is fed into next layer.

4) SOFTMAX LAYER

This layer uses a fully connected layer to adjust the sentiment characteristic of the input layer, and predict tweet sentiment polarity by using Softmax function as follows:

$$y = softmax(\mathcal{M}\hat{c} + b) \tag{17}$$

where $\mathcal{M}$ is a transition matrix of Softmax layer.

The detail of the hyperparameters of the CNN model is presented in TABLE 5.

TABLE 5. Hyperparameters for CNN model.

Hyperparameters	Values
# hidden layer	3
comma-separated filter sizes	5, 3, 2
# filters	128
l_2 regularization	0.0001
# epochs	200
dropout keep probability	0.1
# batch size	100
# k-fold	10

V. EXPERIMENT

A. DATA ACQUISITION

The proposed method was applied to tweet data which has been the subject of an experiment in previous works (DB_1) [26]. The DB_1 dataset is constructed by using the available Python package called Tweepy.[5,6] This dataset was collected by searching all English tweets from Twitter for whole hashtags related to the fuzzy semantic words and the negation words, e.g., #quite, #too, #not, #no, etc. in the period time from May 1^{st}, 2018 to November 30^{th}, 2018 with all topics. Then, 7368 tweets fit our model are selected, divided and stored into two separate database files to use for the experiment as follows: the training data consists of 5158 tweets, and the testing data includes 2210 tweets. Additionally, to prove the performance of our feature ensemble model and to guarantee the fair comparison between our proposed method with other methods, we added 14865 English tweets of the airline companies obtained from the Kaggle website[7] (DB_2). Each original tweet in $DB2$ is assigned one of three kinds of labels, such as "positive," "neutral," and "negative." Therefore, in order to conform with our proposal, the label of tweets in $DB2$ has been reassigned. These tweets are then divided into two separate database files as follows: the training set consists of 10400 tweets, and the testing set includes 4465 tweets. The elements in tweets of both DB_1 and DB_2, such as punctuation marks, re-tweet symbols, URLs, hashtags, and query terms are extracted and removed. Next, a describing text replaces an emoji icon in tweets by using the Python emoji package.[8] In addition, tweets are informal in which users can use acronyms as well as make spelling errors. These can affect the accuracy of the result. Therefore, the Python-based Aspell library[9] is employed to implement spelling corrections. The data was annotated with five labels: *Strong positive*, *Positive*, *Neutral*, *Negative*, and *Strong negative*. We also annotated the testing set as the gold standard to assess the performance. The statistics of these datasets are presented in TABLE 6.

TABLE 6. Statistics of datasets.

Dataset	SP	Positive	Neutral	Negative	SN	Total
DB_1-Train	1129	911	1348	863	907	5158
DB_1-Test	398	442	535	463	372	2210
DB_2-Train	453	2348	817	6382	400	10400
DB_2-Test	126	1013	402	2716	208	4465

where SP = Strong positive, SN = Strong negative.

B. EVALUATION RESULTS

Metrics used to assess the proposed method include *precision, recall*, and F_1. The values of *precision, recall*, and F_1 are computed as follows:

$$Precision = \frac{TP}{TP + FP} \tag{18}$$

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{20}$$

where, TP (True Positive) represents the number of exactly classified items, FP (False Positive) is the number of misclassified items, FN (False Negative) is the number of misclassified non-items.

[5]https://pypi.org/project/tweepy/
[6]https://pypi.org/project/tweepy/
[7]https://www.kaggle.com/crowdflower/twitter-airline-sentiment
[8]https://pypi.org/project/emoji/
[9]https://pypi.org/project/aspell-python-py2/

C. RESULTS AND DISCUSSION

To prove the performance of tweet embeddings created by our feature ensemble model is better than other models; we implemented the same CNN algorithm three times with the input layer formed by three different feature ensemble models. The first time, the vectors created by GloVe model were used (Baseline 1). The second time, vectors generated by the Word2Vec model [20] were employed (Baseline 2), and the third time, vectors were created by our proposed vectors.

TABLES 7, 8, and 9 present the confusion matrix of the proposed, baseline 1, and baseline 2 methods, respectively.

TABLE 7. Confusion matrix of proposed method.

DB1						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	335	37	26		
	Positive	29	390	23		
	Neutral	23	36	351	66	59
	Negative			8	411	44
	SN			8	31	333
DB2						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	91	19	16		
	Positive	40	931	42		
	Neutral	12	21	289	66	14
	Negative			101	2414	201
	SN			12	17	179

where SP = Strong positive, SN = Strong negative.

TABLE 8. Confusion matrix of Baseline 1.

DB1						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	308	31	38	9	12
	Positive	34	327	31	23	27
	Neutral	31	38	331	66	69
	Negative	27	31	22	334	49
	SN	23	13	9	47	280
DB2						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	116	3	1	2	4
	Positive	51	831	46	38	47
	Neutral	1	2	388	4	7
	Negative	104	127	138	2153	194
	SN	1	0	2	4	201

where SP = Strong positive, SN = Strong negative.

In TABLE 7, we can see that the distribution of tweets among sentiments in the dataset is not balanced. Confusion often occurs in the labeling of tweets and assigning labels such as "strong positive," "positive," "neutral," and "strong negative," "negative," and "neutral." For instance, there are 37 tweets in *DB*1 and 19 tweets in *DB*2 misassigned from "strong positive" to "positive," and 26 tweets in *DB*1 and 16 tweets in *DB*2 misclassified from "strong positive" to "neutral" and so on. Generally, there are 17.6% tweets in *DB*1 and 12.6% tweets in *DB*2 misclassified. There is no mislabeling between "strong positive" and "strong negative," or "positive" and "negative," or "strong positive" and "negative," or "strong negative" and "positive." The main reason is that a part of tweets in the training data is not labeled precisely-or the difference among features is unclear. It could also be that the signs to distinguish sentiment among tweets containing these sentiments are quite similar.

TABLE 9. Confusion matrix of Baseline 2.

DB1						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	319	21	20	25	13
	Positive	27	348	21	25	21
	Neutral	49	49	320	57	60
	Negative	27	21	22	365	28
	SN	21	16	17	37	281
DB2						
	Predicted classes					
		SP	Positive	Neutral	Negative	SN
Actual classes	SP	119	1	2	1	3
	Positive	51	832	45	38	47
	Neutral	1	1	396	2	2
	Negative	80	98	125	2310	103
	SN	0	0	2	1	205

where SP = Strong positive, SN = Strong negative.

Using the confusion matrices in TABLE 7, 8, 9 and three metrics (see equations (18), (19), and (20)), the performance of the feature ensemble models was calculated as TABLE 10.

TABLE 10. Comparison of performance of the feature ensemble models.

	DB_1		
	Proposed method	Baseline 1	Baseline 2
Precision	0.81	0.71	0.74
Recall	0.82	0.73	0.75
F_1	0.81	0.72	0.74
	DB_2		
	Proposed method	Baseline 1	Baseline 2
Precision	0.73	0.67	0.72
Recall	0.82	0.89	0.92
F_1	0.76	0.74	0.79

DB_1: a set of tweets containing fuzzy sentiment
DB_2: a set of normal tweets

TABLE 10 shows the accuracy of GloVe, Word2Vec, and our proposed vectors on the CNN model for two datasets presented in Section V.A. As we can see, the proposed method has the highest accuracy, and the GloVe has the lowest accuracy among the three methods. For *DB*1, the proposed method has improved the efficiency of GloVe by up to 9% and Word2Vec by up to 7% for sentiment analysis in tweets containing fuzzy sentiment. For *DB*2, the proposed method has improved the performance of GloVe by 2% for sentiment analysis in tweets. However, the performance of the proposed method is lower than Word2Vec by 3%. According to our assessment, one of the main reasons to achieve this performance is a whole of tweets in *DB*1 containing fuzzy sentiment that is more appropriate for our feature ensemble model than *DB*2. Besides, the elements related to the fuzzy sentiment such as fuzzy semantic words and negation words are extracted and used. In addition, the result shows that the number of tweets also affects the accuracy of the methods.

The more tweets the dataset has, the higher the efficiency is. In general, our proposal applying on *DB*1 achieves better results in comparison to *DB*2 (by 5%). Meanwhile, the accuracy of the GloVe and Word2Vec models increases by 2% and 5% from *DB*1 to *DB*2, respectively. The reason for this is that *DB*2 includes normal tweets, and the Word2Vec and GloVe models are built mainly for classifying tweets containing clearly sentiment. That proves the features ensemble model to treat tweets containing fuzzy sentiment is necessary, and it can improve the performance of sentiment analysis methods.

TABLE 11 shows the performance of the sentiment analysis in tweets contraining fuzzy sentiment.

TABLE 11. Performance of proposed method.

DB_1					
	SP	Positive	Neutral	Negative	SN
TP	335	390	351	411	333
FP	52	73	65	97	103
FN	63	52	184	52	39
Precision	0.87	0.84	0.84	0.81	0.76
Recall	0.84	0.88	0.66	0.89	0.90
F_1	0.85	0.86	0.74	0.85	0.82
DB_2					
	SP	Positive	Neutral	Negative	SN
TP	91	931	289	2414	179
FP	52	40	171	83	215
FN	35	82	113	302	29
Precision	0.64	0.96	0.63	0.97	0.45
Recall	0.72	0.92	0.72	0.89	0.86
F_1	0.68	0.94	0.67	0.93	0.60

where SP = Strong positive, SN = Strong negative.

From TABLE 11, it can be seen that for *DB*1, the "strong positive" and "positive" and "negative" classes have been classified better than the remaining ones. Intuitively, one of the main reasons for the low performance is that the training data contains fewer tweets indicating "strong negative" and "neutral" sentiments. Meanwhile, for *DB*2, the "positive" and "negative" classes have been classified better than the "strong positive" and "strong negative" and "neutral" classes. The main reason is that most of the tweets in *DB*2 contain not so many tweets containing fuzzy sentiment as *DB*1. Therefore, the number of tweets labeled "strong positive" and "strong negative" and "neutral" in *DB*2-Train is very low. We believe that with the construction of a large data warehouse and a better balance between tweets indicating relevant factors, this result can be significantly improved.

The sentiment analysis effectiveness of the proposed method and the baseline method is shown in TABLE 12. In which, the baseline method is our other study which is published as the conference paper (Baseline 3) [26]. For a fair comparison, the methods are implemented on the same dataset and parameters.

From TABLE 12, the average result of the methods is further clarified by the data in TABLE 13.

According to TABLE 13, the proposed method obtains better results than the baseline method. Although the disparity in performance is not so high, it proves that this study can still improve the accuracy of analyzing the sentiment of tweets containing fuzzy sentiment by up to 9% compared to the baseline method. Why can the proposed method improve the accuracy of the baseline method? In this paper, the tweet embeddings are built by using the information related to the lexicon, word-type, semantic, position, and polarity sentiment of words. Furthermore, the sentiment score of fuzzy semantic words and fundamental words are calculated more precisely. In addition, the CNN algorithm used to classify the sentiment of tweets is one of the algorithms that achieve good accuracy for analyzing sentiment at the moment. The results again confirm that building tweet embeddings has a significant impact on the accuracy of the sentiment analysis methods.

TABLE 12. Comparison of performance of sentiment analysis methods on *DB*1.

	Proposed method			Baseline 3 method		
	Precision	Recall	F_1	Precision	Recall	F_1
SP	0.87	0.84	0.85	0.65	0.71	0.68
Positive	0.84	0.88	0.86	0.83	0.78	0.80
Neutral	0.84	0.66	0.74	0.71	0.67	0.69
Negative	0.81	0.89	0.85	0.74	0.85	0.79
SN	0.76	0.90	0.82	0.64	0.61	0.63

where SP = Strong positive, SN = Strong negative.

TABLE 13. Average of performance of sentiment analysis methods on *DB*1.

	Proposed method	Baseline 3
Precision	0.81	0.71
Recall	0.82	0.72
F_1	0.81	0.72

VI. CONCLUSION AND FUTURE WORK

This work proposed a method for improving the performance of sentiment analysis in tweets containing fuzzy sentiment based on the feature ensemble and CNN models. The feature ensemble model was built by concatenating information from five feature vectors extracted from lexical, word-type, semantic, sentiment polarity, and position of words in tweets containing fuzzy sentiment phrases. The result obtained using this model is tweet embeddings, which was used as feature vectors in the input layer of the CNN model. The experiment analysis revealed that the proposed method significantly improved the performance in the sentiment analysis of tweets containing fuzzy sentiment. There are some possible limitations of the proposed approach: the method only considered tweets containing fuzzy sentiment without considering the influence of other elements in them such as slang and sarcasm. In the future, we plan to analyze the sentiment of tweets by considering other information using the BERT model for tweets.

APPENDIXES

APPENDIX A

LIST OF NEGATION WORDS

See TABLE 14.

APPENDIX B

TABLE 14. List of negation words.

Order	Word	Order	Word	Order	Word	Order	Word
1	aint	11	hasnt	21	neither	31	n't
2	cannot	12	havent	22	never	32	n't
3	cant	13	havnt	23	no	33	nt
4	darent	14	isnt	24	nobody	34	oughtnt
5	denied	15	lack	25	none	35	hant
6	denies	16	lacking	26	noone	36	shouldnt
7	didnt	17	lacks	27	nor	37	wasnt
8	doesnt	18	mightnt	28	not	38	without
9	dont	19	mustnt	29	nothing	39	wouldnt
10	hadnt	20	neednt	30	nowhere		

TABLE 15. List of diminisher words.

Word	Score	Word	Score	Word	Score
barely	-0.09	little	-0.1	reasonably	-0.09
fairly	-0.05	moderately	-0.03	scanty	-0.06
few	-0.06	partly	-0.07	scarcely	-0.06
bit	-0.05	partially	-0.07	slightly	-0.11
hardly	-0.18	quite	-0.05	some	-0.08
insignificantly	-0.12	rarely	-0.1	somewhat	-0.08
less	-0.13	relatively	-0.01	sparsely	-0.08
				tolerably	-0.09

LIST OF DIMINISHER WORDS

See TABLE 15.

APPENDIX C
LIST OF INTENSIFIER WORDS

See TABLE 16.

TABLE 16. List of intensifier words.

Word	Score	Word	Score	Word	Score
absolutely	0.25	largely	0.24	sure	0.10
all	0.25	literally	0.24	surely	0.10
altogether	0.25	most	0.24	surprisingly	0.25
almost	0.24	perfectly	0.25	thoroughly	0.25
completely	0.25	pretty	0.01	totally	0.18
enormously	0.23	purely	0.25	too	0.11
enough	0.24	rather	0.1	truly	0.07
entirely	0.25	real	0.12	utterly	0.25
exceedingly	0.25	really	0.04	very	0.09
extremely	0.2	so	0.055	virtually	0.07
highly	0.24	strongly	0.24	whole	0.24
				wholly	0.24

REFERENCES

[1] O. Araque, I. Corcuera-Platas, J. F. Sánchez-Rada, and C. A. Iglesias, "Enhancing deep learning sentiment analysis with ensemble techniques in social applications," *Expert Syst. Appl.*, vol. 77, no. 19, pp. 236–246, Jul. 2017.

[2] N. Al-Twairesh, and H. Al-Negheimish, "Surface and deep features ensemble for sentiment analysis of arabic tweets," *IEEE Access*, vol. 7, pp. 84122–84131, 2019.

[3] M. Z. Asghar, A. Khan, S. Ahmad, M. Qasim, and I. A. Khan, "Lexicon-enhanced sentiment analysis framework using rule-based classification scheme," *PLoS ONE*, vol. 12, no. 2, Feb. 2017, Art. no. e0171649.

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

[5] C. Badica and G. Vladutu, "Application of meaningful text analytics to online product reviews," in *Proc. 20th Int. Symp. Symbolic Numeric Algorithms Sci. Comput. (SYNASC)*, Sep. 2018.

[6] G. Bello, H. Menéndez, S. Okazaki, and D. Camacho, "Extracting collective trends from twitter using social-based data mining," in *Proc. Int. Conf. Comput. Collective Intell.* Berlin, Germany: Springer, 2013, pp. 622–630.

[7] S. Bird and E. Loper, "NLTK: The natural language toolkit," in *Proc. Interact. Poster Demonstration Sessions, Assoc. Comput. Linguistics*, 2004, p. 31.

[8] C. Cerisara, P. Kral, and L. L. Lenc, "On the effects of using word2vec representations in neural networks for dialogue act recognition," *Comput. Speech Lang.*, vol. 47, pp. 175–193, Jan. 2018.

[9] H. H. Dohaiha, P. W. C. Prasad, A. Maag, and A. Alsadoon, "Deep learning for aspect-based sentiment analysis: A comparative review," *Expert Syst. Appl.*, vol. 118, pp. 272–299, Mar. 2019.

[10] D. I. H. Farías, E. Sulis, V. Patti, G. Ruffo, and C. Bosco, "Valento: Sentiment analysis of figurative language tweets with irony and sarcasm," in *Proc. 9th Int. Workshop Semantic Eval., SemEvalNAACL-HLT*, Denver, CO, USA, Jun. 2015, pp. 694–698. [Online]. Available: http://aclweb.org/anthology/S/S15/S15-2117.pdf

[11] M. Ganapathibhotla and B. Liu, "Mining opinions in comparative sentences," in *Proc. 22nd Int. Conf. Comput. Linguistics Conf.*, 2008, pp. 241–248.

[12] J. Gehring, M. Auli, D. Grangier, D. Yarats, Y. N. Dauphin, "Convolutional sequence to sequence learning," 2017, *arXiv:1705.03122*. [Online]. Available: https://arxiv.org/abs/1705.03122

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

[14] M. Kamkarhaghighi and M. Makrehchi, "Content tree word embedding for document representation," *Expert Syst. Appl.*, vol. 90, pp. 241–249, Dec. 2017.

[15] A. Kennedy and D. Inkpen, "Sentiment classification of movie reviews using contextual valence shifters," *Comput. Intell.*, vol. 22, no. 2, pp. 110–125, 2006.

[16] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*. [Online]. Available: https://arxiv.org/abs/1408.5882

[17] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 971–980.

[18] F. M. Kundi, S. Ahmad, A. Khan, and M. Z. Asghar, "Detection and scoring of Internet slangs for sentiment analysis using SentiWordNet," *Life Sci. J.*, vol. 11, no. 9, pp. 66–72, 2014.

[19] A. Hassan and A. Mahmood, "Convolutional recurrent deep learning model for sentence classification," *IEEE Access*, vol. 6, pp. 13949–13957, 2018.

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: https://arxiv.org/abs/1301.3781

[21] S. M. Mohammad, S. Kiritchenko, and X. Zhu, "NRC-Canada: Building the state-of-the art in sentiment analysis of tweets," 2013, *arXiv:1308.6242*. [Online]. Available: https://arxiv.org/abs/1308.6242

[22] R. Narayanan, B. Liu, and A. Choudhary, "Sentiment analysis of conditional sentences," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, vol. 1, 2009, pp. 180–189.

[23] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep convolution neural networks for twitter sentiment analysis," *IEEE Access*, vol. 6, pp. 23253–23260, 2018.

[24] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.

[25] H. T. Phan, N. T. Nguyen, V. C. Tran, and D. Hwang, "A method for detecting and analyzing the sentiment of tweets containing conditional sentences," in *Proc. 11th Asian Conf. Intell. Inf. Database Syst.* Cham, Switzerland: Springer, Apr. 2019, pp. 177–188.

[26] H. T. Phan, N. T. Nguyen, T. Van Cuong, and D. Hwang, "A method for detecting and analyzing the sentiment of tweets containing fuzzy sentiment phrases," in *Proc. IEEE Int. Symp. Innov. Intell. Syst. Appl. (INISTA)*, Jul. 2019, pp. 1–6.

[27] Y. Ren, R. Wang, and D. Ji, "A topic-enhanced word embedding for Twitter sentiment classification," *Inf. Sci.*, vol. 369, pp. 188–198, Nov. 2016.

[28] A. U. Rehman *et al.*, "A hybrid CNN-LSTM model for improving accuracy of movie reviews sentiment analysis," *Multimedia Tools Appl.*, vol. 78, pp. 26597–26613, Jun. 2019, doi: 10.1007/s11042-019-07788-7.

[29] S. M. Rezaeinia, R. Rahmani, A. Ghodsi, and H. Veisi, "Sentiment analysis based on improved pre-trained word embeddings," *Expert Syst. Appl.*, vol. 117, pp. 139–147, Mar. 2019.
[30] G. Sidorov, F. Velasquez, E. Stamatatos, A. Gelbukh, and L. Chanona-Hernández, "Syntactic N-grams as machine learning features for natural language processing," *Expert Syst. Appl.*, vol. 41, no. 3, pp. 853–860, Feb. 2014.
[31] F. Strohm, "The impact of intensifiers, diminishers and negations on emotion expressions," M.S. thesis, Univ. Stuttgart, Stuttgart, Germany, 2017.
[32] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguistics*, vol. 37, no. 2, pp. 267–307, Jun. 2011.
[33] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, and B. Qin, "Learning sentiment-specific word embedding for Twitter sentiment classification," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2014, pp. 1555–1565.
[34] H. Thakkar and D. Patel, "Approaches for sentiment analysis on Twitter: A state-of-art study," 2015, *arXiv:1512.01043*. [Online]. Available: https://arxiv.org/abs/1512.01043
[35] H. T. Phan, N. T. Nguyen, V. C. Tran, and D. Hwang, "A sentiment analysis method of objects by integrating sentiments from tweets," *IFS*, vol. 37, no. 6, pp. 7251–7263, Dec. 2019.
[36] Z. Ye, F. Li, and T. Baldwin, "Encoding sentiment information into word vectors for sentiment analysis," in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 997–1007.

HUYEN TRANG PHAN received the M.S. degree in computer science from the University of Science and Technology, The University of Da Nang, Vietnam, in 2015. She is currently pursuing the Ph.D. degree with the Department of Computer Engineering, Yeungnam University, South Korea. She has authored one journal article and five conference papers. Her research interests include text summarization, sentiment analysis, decision support systems, machine learning, and deep learning.

VAN CUONG TRAN was born in Vietnam. He received the B.S. degree in computer science from the Hue University of Sciences, Vietnam, in 2012, and Ph.D. degree in computer engineering from Yeungnam University, South Korea, in 2017. He is currently a Professor with Quang Binh University, Vietnam. He has the authored seven journal articles and nine conference papers. His researches have focused on named entity recognition, sentiment analysis, and recommendation systems.

NGOC THANH NGUYEN (Senior Member, IEEE) is currently a Full Professor with the Wroclaw University of Science and Technology and the Head of Information Systems Department, Faculty of Computer Science and Management. He is also the Honorary Chair of the Scientific Board with Nguyen Tat Thanh University. His scientific interests consist of collective intelligence, knowledge integration methods, inconsistent knowledge processing, and multiagent systems. He has edited more than 30 special issues in international journals, 52 books, and 35 conference proceedings. He has authored or coauthored of 5 monographs and more than 350 journal and conference papers. He serves as an Editor-in-Chief of the *International Journal of Information and Telecommunication* (Taylor&Francis), the *Transactions on Computational Collective Intelligence* (Springer), and *Vietnam Journal of Computer Science* (World Scientific). He is also an Associate Editor-in-Chief of several prestigious international journals, among others, the *Journal of Intelligent and Fuzzy Systems*, *Applied Intelligence*. He was a General Chair or Program Chair of more than 40 international conferences. He serves as a member of the Council of Scientific Excellenceof Poland, a member of Committee on Informatics of the Polish Academy of Sciences, an Expert of National Center of Research and Development and European Commission in evaluating research projects in several programs like Marie Sklodowska-Curie Individual Fellowships, FET and EUREKA. He has given 22 plenary and keynote speeches for international conferences, and more than 40 invited lectures in many countries. In 2009, he was granted of title Distinguished Scientist of ACM. He was also a Distinguished Visitor of the IEEE and a Distinguished Speaker of ACM. He also serves as the Chair for IEEE SMC Technical Committee on Computational Collective Intelligence.

DOSAM HWANG received the Ph.D. degree from Kyoto University, Kyoto, Japan. He is currently a Full Professor with the Department of Computer Engineering, Yeungnam University, South Korea. His research interests mainly include natural language processing, ontology, knowledge engineering, information retrieval, and machine translation. He has served as the Head of the Yeungnam University's Computer Engineering Department for five years from 2005 to 2009. He has also held a position as a Principal Researcher with the Korea Institute of Science and Technology (KIST) and has also been a Visiting Professor with the Korea Advanced Institute of Science and Technology (KAIST). He has so far been not only a co-chair of several international conferences but also a steering committee member of ICCCI and ACIIDS, and MISSI international conferences. More specifically, for example, he has been the Assistant Secretary of ISO/TC37/SC4 for language resource management from 2005 to 2007, where he is also the Secretary of Korean TC for ISO/TC37/SC4. In 2006, he was the Director of the Korean Society for Cognitive Science (KSCS) and the Korean Information Science Society (KISS). He has been serving as the Society's Director and the Mentor of the Knowledge Engineering Study Group, since 2007. In addition to this, he has also participated in several Korean National Research Projects, such as a project on machine translation system (from 1985 to 1990), and the National IT Ontology Infrastructure and Technology Development Project called CoreOnto (2006–2009), and Exobrain, (2013–2014), the Project focused on the Construction of Deep Knowledge Base and Question-Answering Platform. He has been the In Charge of an intelligent service integration based on IoT Big Data as part of Korea's another principal national research project BK+ since 2014. In recognition of his such great commitment and contribution to the relative fields of study, he has been honored as a Distinguished Researcher of KIST in 1988 by Korea's Ministry of Science and Technology (MoST) and awarded a prize for Good Conduct from Kyunghee High School in 1973. He had more than 50 publications.