Refining Word Embeddings with Sentiment Information for Sentiment Analysis

Mohammed Kasri[1,*], Marouane Birjali[1], Mohamed Nabil[1], Abderrahim Beni-Hssane[1], Anas El-Ansari[2] and Mohamed El Fissaoui[2]

[1]*Computer Science Department, Chouaib Doukkali University, Faculty of Sciences, El Jadida, Morocco*
[2]*Mohammed First University, MASI Laboratory, Nador, Morocco*
E-mail: kasri.m@ucd.ac.ma; birjali.marouane@gmail.com; nabilmed77@gmail.com; abenihssane@yahoo.fr; anas.elansari@gmail.com; mohamed.el.fissaoui@gmail.com
[*]*Corresponding Author*

Received 08 March 2022; Accepted 30 March 2022;
Publication 10 August 2022

Abstract

Natural Language Processing problems generally require the use of pre-trained distributed word representations to be solved with deep learning models. However, distributed representations usually rely on contextual information which prevents them from learning all the important word characteristics. The task of sentiment analysis suffers from such a problem because sentiment information is ignored during the process of learning word embeddings. The performance of sentiment analysis can be affected since two words with similar vectors may have opposite sentiment orientations. The present paper introduces a novel model called Continuous Sentiment Contextualized Vectors (CSCV) to address this problem. The proposed model can learn word sentiment embedding using its surrounding context words.

Journal of ICT Standardization, Vol. 10_3, 353–382.
doi: 10.13052/jicts2245-800X.1031

It uses Continuous Bag-of-Words (CBOW) model to deal with the context and sentiment lexicons to identify sentiment. Existing pre-trained vectors are combined then with the obtained sentiment vectors using Principal component analysis (PCA) to enhance their quality. The experiments show that: (1) CSCV vectors can be used to enhance any pre-trained word vectors; (2) The result vectors strongly alleviate the problem of similar words with opposite polarities; (3) The performance of sentiment classification is improved by applying this approach.

Keywords: Sentiment embeddings, sentiment analysis, word embeddings, sentiment lexicon, deep learning.

1 Introduction

Distributed representations or word embeddings are learned representations for a qualitative concept (e.g., word, document), where close concepts usually have a similar representation [1]. As the name indicates, these techniques distribute the information about a given concept all along the vector. Word embedding techniques train neural network or vector space models [2] with a large corpus to obtain the corresponding vector for each concept. One of the important advantages of distributed representations is the ability to capture the syntactic and semantic relations between words. The performance of natural language processing (NLP) tasks including question answering [3] sentiment analysis (SA), has significantly improved due to word embedding and deep learning (DL) models [4–7]. In NLP tasks, DL models exploit word embedding to automatically find and extract high-level features from textual data [8]. On the other hand, since most word embedding models rely on context, the extracted features are more related to syntactic and semantic orientation.

The two successful word embedding methods are word2vec [9, 10] and Global Vectors (GloVe) [11]. Word2Vec model includes two learning algorithms: (1) Continuous Bag-of-Words (CBOW) which aims to predict a word given its context, and (2) Skip-Gram whose goal is to predict the context given a word. The difference between these two algorithms is that CBOW is faster to train, and can predict frequent words more accurately, while Skip-gram performs well for a small amount of data and can be slightly accurate for rare words. GloVe instead is a global log-bilinear regression model that obtains the vectors using co-occurrence statistics and

matrix factorization. More word embeddings models proposed later such as FastText [12], Elmo [13], Bert [14], and GPT-2 [15].

Although distributed representations have improved the performance of NLP tasks, they have some drawbacks. For example, most word embedding models require a large corpus [16, 17]. This is because they rely on contextual information and hence a large corpus will help these models to obtain more acceptable word vectors [18]. The problem with these models is they extract all the important characteristics of a word from its surrounding context words [19]. As a result, the obtained word vectors contain only semantic and syntactic information. Moreover, if two words have similar linguistic features, they will be located close to each other in the vector space because they have appeared several times in a common context within the corpus [20]. In some NLP tasks like information retrieval [21, 22], this situation may not be a problem. However, the performance of some NLP tasks such as sentiment analysis will be affected by this fact, because in sentiment analysis we do not rely only on contextual information to determine the sentence sentiment [23, 24]. Since word embedding models ignore sentiment information, the closest words in the vector space may have opposite sentiment polarities. For instance, although the two words "good" and "bad" have extreme opposite sentiments, word2vec consider them 71% similar. The work of [25] shows that about 30% of the top 10 semantically similar words include at least one word with opposite sentiment polarity. This may reduce the efficiency of word embedding for sentiment analysis and affect the performance of this task. This study addressed this problem to provide high-quality word embeddings for sentiment analysis.

More specifically, we aim to answer the following questions: (1) can the sentiment polarity of a word be used directly in word embeddings models to learn sentiment information? and (2) how the integration of such important features into existing pre-trained word embeddings may affect the performance of SA. In this task, we propose a model named Continuous Sentiment Contextualized Vectors (CSCV) to capture sentiment features based on the CBOW algorithm and sentiment lexicons. In our approach, the CBOW model is modified to predict sentiment from surrounding context words. In particular, the target word will be treated by its sentiment polarity (strong positive, positive, neutral, negative, and strong negative) based on three well-known lexicons, namely SentiWordNet [26], SenticNet [27], and VADER [28]. The advantage of this approach is generating a word vector that contains sentiment information in addition to some semantic and syntactic regularities. Thus, the

sentiment vectors can be combined with other pre-trained vectors to generate high-quality word embeddings. The key idea is to have word vectors with all necessary information to distinguish between words with similar contexts and opposite sentiment polarities.

This study is an extension of our previous work: Enhanced Word Embeddings with Sentiment Contextualized Vectors for Sentiment Analysis [29]. The main goal of this study is to propose a novel model called Continuous Sentiment Contextualized Vectors (CSCV). This model aims to learn sentiment embeddings by making use of sentiment lexicons and the CBOW model. The obtained results show that combining these sentiment embeddings with other context-based word embeddings improved the performance of sentiment analysis using deep learning approaches. The rest of this paper is organized as follows: we review related work in Section 2. The proposed sentiment embedding model for SA is presented in Section 3. Section 4 elaborates on the experiment setup, evaluation results, and discussion. Finally, Section 5 concludes the previous results and outlines future works.

2 Related Work

2.1 Word Embedding Methods

Word embedding aims to learn distributed vector representations of words by exploiting the existence of a huge amount of contextual information in a large text corpus using different techniques. This idea starts with [30]. They proposed a neural network language model that learns word embeddings by predicting each word based on its preceding contexts. Because this model is not based on a fixed-size context window, it requires more computation. This problem was solved by the C&W model proposed by [31, 32] which learns word representation using a binary classification task. The goal was to determine if the word in the middle is related to its preceding and succeeding context words using Convolutional Neural Network (CNN) architecture. [9, 10] proposed the word2Vec method with two shallow neural network models for word representation namely Continuous Bag-of-Words (CBOW) and Skip-Gram. CBOW learns to predict a word from its surrounding context, or maximize the probability of the target word by looking at the context. On the other hand, the skip-gram model is designed to predict the context from of a given word. Those methods do not properly make use of corpus global statistics, because they rely on linear and local contexts (fixed-size context window before and after a word). Unlike the previous

methods, [33] proposed dependency-based word embeddings to overcome the problem of linear contexts. They generalized the Skip-Gram model by replacing the linear bag-of-words contexts with arbitrary ones. To address the problem of local context, [11] proposed Global Vector (GloVe) model that learns word embeddings based on the global statistical information of words in the corpus. The model constructs a word-word co-occurrence matrix to make a projection between words vectors and the conditional probability of their corresponding words. (Devlin et al., 2018) proposed a simple and powerful language representation model called Bidirectional Encoder Representations from Transformers (BERT) to improve the performance of NLP tasks. Another model called generative pre-training (GPT) is proposed by [34] and its successor GPT-2 [15] to learn a universal representation that transfers with little adaptation.

Many word embedding approaches have been proposed in the literature to embed more useful and complex information in the word vectors. Contextual information proved their effectiveness in learning word embeddings. However, [12] show that character-level sub-words can be used also to obtain word vectors. They proposed an extension of the Skip-Gram model where each word is represented as a bag of character n-gram to obtain morphological information of words. The advantage of this method is the ability to compute word representations for words that did not appear in the training corpus. [13] proposed a new type of deep contextualized word representation to model the polysemy in addition to semantic and syntactic regularities. Part-of-Speech (POS) information can be introduced into a neural network language model. In this regard, [35] proposed the Continuous Dissociation between Nouns and Verbs Model (CDNV) which is similar to CBOW model. They exploited the principle of the dissociation between nouns and verbs (DNV) characteristic in language acquisition to improve the quality of word embeddings. Although these models have improved significantly the performance of various NLP tasks, they generally ignore sentiment information. We address this problem by proposing a sentiment embeddings method to avoid generating similar vector representations for sentimentally opposite words, which is certainly will be useful for SA task.

2.2 Sentiment Embedding Methods

There are two common methods to integrate sentiment information into word vectors: The first method learns sentiment information through labeled (polarity labels) corpora while the second method makes use of sentiment

lexicons to enhance existing pre-trained word embedding. The two methods have shown a considerable improvement in the sentiment analysis task. [36] adopted the first method and proposed a semi-supervised method to integrate both sentiment and semantic characteristics. Their approach used two models; the first one was dedicated to capturing semantic information based on a probabilistic topic technique. On the other hand, the second model uses logistic regression to perform sentiment classification. Moreover, this model exploited this phase to learn sentiment embeddings from the labeled corpus. They have maximized the sum of their objective function to unify the two models.

Several studies have developed network models to capture sentiment embeddings from tweets during the process of Twitter sentiment classification. [37] presented Sentiment Specific Word Embedding (SSWE) method for SA. It aims to encode sentiment information in the continuous representation of words. The authors used a multitask neural network model like the one proposed by [31], but to unify the two models, they used a weighted sum of individual loss functions. Similarly, [38] developed a neural network model based on [31, 37] methods to learn Multi-prototype Topic and Sentiment-enriched Word Embeddings (M-TSWE). [39] proposed three CNN-based models to capture sentiment, semantic and syntactic information and then integrates them to generate sentiment word vectors (SWV) in three different strategies, namely combined (SWV-C), mixed (SWV-M), and hybrid (SWV-H).

Since the first approach relies on machine learning algorithms, it requires labeled data that is not always available. Moreover, the models adopting this approach are complex and they need the tunning of multiple parameters. As a result, several studies exploited sentiment lexicons to refine existing word vectors. [25] used this approach based on the intensity score of the extended version of Affective Norms of English Words (E-ANEW) [40] to enhance word embedding. Similarly, [41] introduced a model called Sentiment-augmented Convolutional Neural Networks (SCNN). The authors designed this model to contain two input and lookup layers. The first lookup layer uses vectors from word2vec. The second one learns sentiment embedding using SentiWordNet(SWN) sentiment lexicon [26]. The work of [42], proposed a word vector refinement model which can be applied to existing pre-trained word vectors. The proposed model aims to adjust word representations to be closer semantically and sentimentally and thus improve the performance of word embeddings for sentiment analysis. This work differs from our work as it calculates the cosine similarity between a given word and other words

in a sentiment lexicon. Next, it selects the top-k most similar words since they represent the closest neighbors and r-rank them based on their sentiment scores in the lexicon. Finally, the word vectors are refined according to this rank so they are closer semantically and sentimentally. However, in this study, we train an extension of the CBOW model to predict sentiments -obtained from lexicons- from surrounding context words. [24] Proposed Improved Word Vectors (IWV) to refine pre-trained word embedding. This method aims to improve the accuracy of sentiment analysis using deep learning. The authors tried to combine multiple information such as POS tag and word sentiment into IWV. [23] used also E-ANEW lexicon in addition to pre-trained vectors to predict the word polarity based on a neural network model. The authors exploited the obtained vectors to ameliorate the performance of sentiment analysis

3 Learning Sentiment Embeddings

As CSCV learns sentiment embeddings from surrounding contexts, it follows a neural network architecture similar to CBOW model. Therefore, we begin this section with a brief description of CBOW, and then present in detail our proposed approach. The construction of CSCV imposed two changes on CBOW model as follows:

- In place of predicting a target word based on its local context, the model exploits the local context of this word to predict its sentiment polarity (e.g., positive or negative).
- Several parameters have been added or modified to make the model generates high-quality vectors.

3.1 Continuous Bag-of-Words (CBOW) Model

CBOW model is a feed-forward neural network with three layers, namely input, projection, and output layer as illustrated in Figure 2. This architecture was proposed by [9] and aims to predict a target word (a word of interest or a center word) based on its surrounding context words. Considering this example, "the best camera you can buy today" and a context window (number of words to consider left and right the target word) of size 2, we can construct pairs of (context words, target word) from the text as shown in Figure 1.

At the input layer, a set of one-hot encoded vectors of the context words is given, where the vector size is equal to the vocabulary (unique words in the corpus) size. Then the embedding that corresponds to each word in the

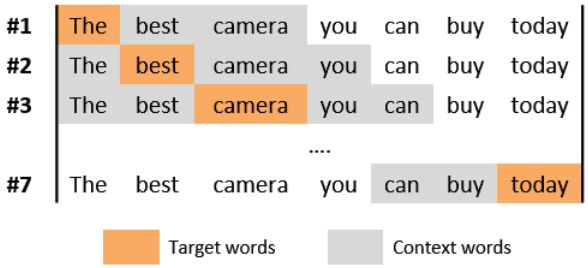

Figure 1 Examples of pairs (context words, target word).

context is obtained from the embedding matrix (lookup table). This matrix is initialized randomly at the beginning by N-dimension dense float vectors. At the projection layer, the embeddings of context words are summed up using bag-of-words and then sent to an output layer which gives the probabilities of each word in the vocabulary being a target word. Thus, SoftMax is used as an activation function for the last layer. The authors adopted hierarchical SoftMax to reduce the computational complexity of the model.

3.2 Sentiment Lexicon

The sentiment prediction from surrounding context words requires the implication of sentiment lexicon and pairs of «context window, target sentiment». We have changed the target word in CBOW model by the sentiment of this word. We have chosen five categories of sentiment namely Negative, Strong Negative, Neutral, Positive, and Strong Positive. The sentiment of each word is identified using a combination of three well-known lexicons. These lexicons are: (1) SentiWordNet [26], (2) SenticNet [27], and (3) VADER [28].

SentiWordNet uses a sentiment score range between 0.0 and 1.0 to assign words' polarity. It groups 117 658 words into synonym sets (syensets) based on the famous lexical resource WordNet [43] where each synset can have three(positive, negative, and objective (usually neutral)) sentiment scores. SenticNet contains about 15 000 words with a sentiment score between -1 and 1. -1 means the word is extremely negative while 1 represents extreme positive words. This lexicon was created to deal with concept-level sentiment analysis. Vader (Valence Aware Dictionary for Sentiment Reasoning) on the other hand represents a rule-based tool and a sentiment lexicon. It was developed for social media sentiment analysis. This lexicon includes about 7 517 words with their polarity scores. In this study, we used SenticNet and VADER to identify the sentiment of words that do not appear in SentiWordNet.

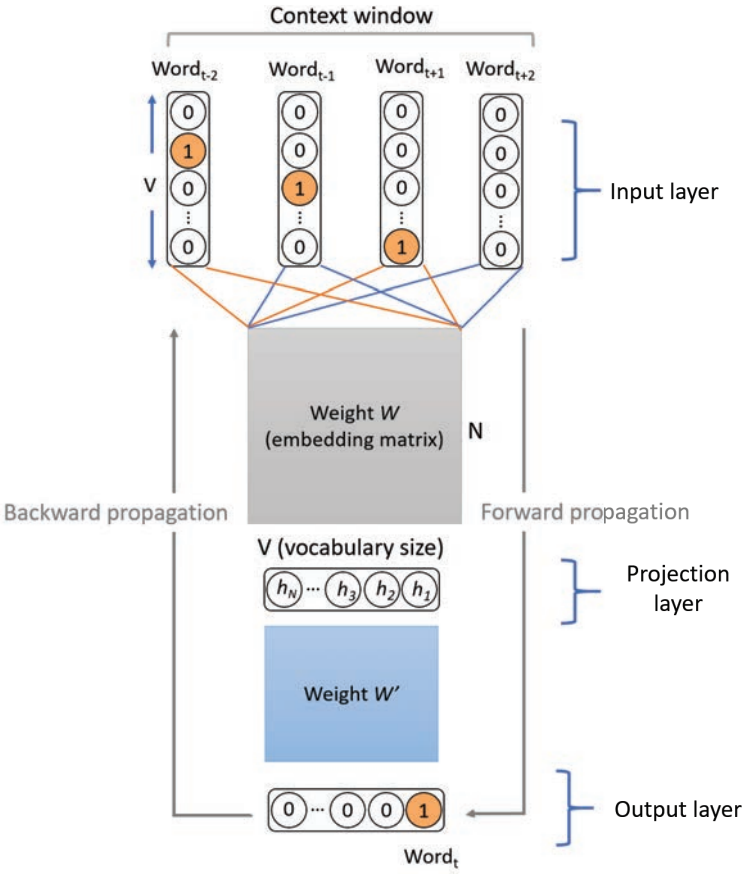

Figure 2 The architecture of CBOW. N denotes the size of word embedding. V denotes the size of the vocabulary.

As summarized in Algorithm 1, The target word is replaced by its sentiment score value from SenticNet and VADER. On the other hand, since the word may appear in different synsets in SentiWordNet, we used the difference between the average of positive and negative scores. This is indicated by the following formula:

$$Score = \frac{\sum_{i=1}^{k} Synset_{p(i)}}{k} - \frac{\sum_{i=1}^{k} Synset_{n(i)}}{k} \tag{1}$$

where k refers to the number of appearances of the word, $Synset_{p(i)}$ represents the positive score of synset i and $Synset_{n(i)}$ represents the negative

score of synset i. The process of finding sentiment scores starts by looking at the largest lexicon. If the word does not appear, it moves to the next lexicon. The words that do not exist in all lexicons are after this, a sentiment class label is assigned to the word based on its obtained score range.

Algorithm 1 Obtaining the sentiment polarity of each word in the vocabulary

Input: A set of words in Vocabulary $= (W_1, W_2, W_3, \ldots, W_V)$
Output: A dictionary of input words and their corresponding sentiment polarity

```
// Get sentiment polarity of words based on SentiWordNet, SenticNet, and VADER lexicons
Sentiment_class = {} // Dictionary of words and their corresponding polarity class
for each W in Vocabulary:
    Score = 0
    Word = Lemmatization(W) //Lemmatize the word
    Synsets = GetSynsets_SentiWordNet(Word) // Obtain the sysnets of a word from SentiWordNet
    if length (Synsets) > 0 do // That means the word exists in SentiWordNet lexicon
        Score = Average( Synsets.positive_scores) - Average( Synsets.negative_scores)
    else
        Score = getPolarity_SenticNet(Word)
        if Not Score do // That means the word does not exist in the SenticNet lexicon
            Score = getPolarity_VADER(Word)
        end if
    end if
    // Assign sentiment class based on the score obtained before
    append (W: "Strong Negative") to Sentiment_class if Score <= -0.5
    append (W: "Negative") to Sentiment_class if -0.5 < Score < 0
    append (W: "Neutral") to Sentiment_class if Score == 0
    append (W: "Positive") to Sentiment_class if 0 < Score < 0.5
    append (W: "Strong Positive") to Sentiment_class if Score >= 0.5
end for
    return Sentiment_class
```

In our approach, we considered the words not present in the lexicons as neutral. This is not always true, because some words may have a positive or negative sentiment polarity. The solution to this problem is using a large lexicon. Unfortunately, available lexicons do not include all the words in the vocabulary set of a required corpus to train word embedding models. Manually generated lexicons are the best resources for such problems, but most of them have small sizes. Therefore, to partially relieve this problem, multiple lexicons can be combined as in this study. However, less accurate lexicons may hurt the quality of word vectors obtained. Thus, we adopt three well-known lexicons that proved their effectiveness in the field of SA.

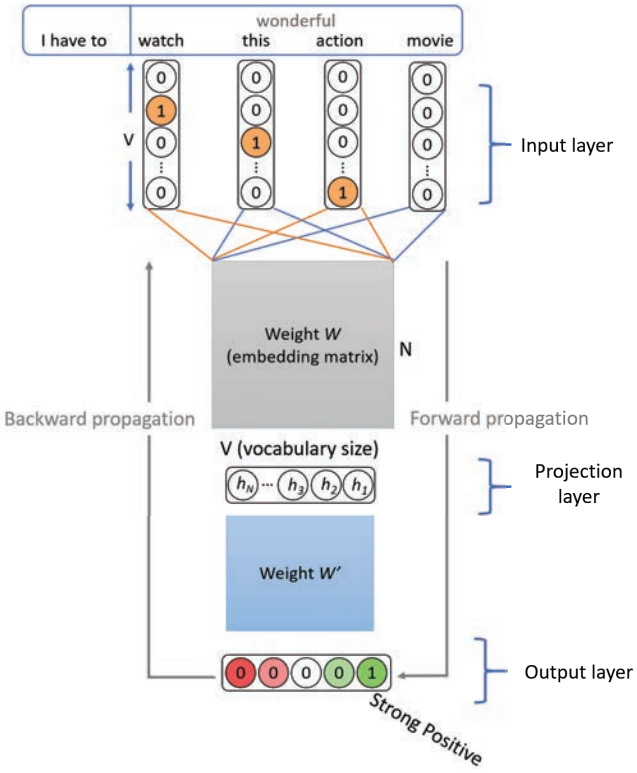

Figure 3 The architecture of CSCV with an example of a strong positive word.

3.3 CSCV Model

The proposed model differs from CBOW model as it changes the target word by its sentiment category class. As a result, instead of learning some semantic regularities of a language, it learns sentiment embedding based on the context words. Figure 3 shows the architecture of CSCV. It consists of a simple neural network with three layers. The words of local context $\mathrm{w_{t-c}}, \ldots, \mathrm{w_{t+c}}$ are encoded to one-hot encoding before they are mapped to their unique vectors $\mathrm{v_{t-c}}, \ldots, \mathrm{v_{t+c}}$ in the projection layer respectively. $\mathrm{w_t}$ is excluded from the local context because it represents the target word for a context window of size c. These vectors are averaged to construct the input $\mathrm{h_t}$. Then $\mathrm{h_t}$ will be passed to the output layer to predict the sentiment polarity (strong positive, positive, neutral, negative, and strong negative) $\mathrm{S_{w_t}}$ of the word $\mathrm{w_t}$.

The input h_t is calculated using the following equation:

$$h_t = \frac{1}{\mathrm{C}} \mathrm{W}^{\mathrm{T}} \cdot \sum_{i=1}^{C} w_i \tag{2}$$

where w_i is the one-hot encoded vector of word i in the context and $\mathrm{W} \in \mathbb{R}^{V \times N}$ represents the embedding matrix (lookup table) of size V (vocabulary size) $\times N$ embedding size. $\mathrm{W}' \in \mathbb{R}^{N \times SC}$ is a weight matrix between the projection and the output layer. SC denotes the number of sentiment polarity (classes) in the output layer. Thus, we can compute the conditional probability for each node in the output layer using the following equation:

$$u_j = v_j^T h_t \tag{3}$$

where $\mathrm{v}_{\mathrm{j}}^{\mathrm{T}}$ is j^{th} column of matrix W'. These values are then passed through a Softmax activation function to predict the most acceptable sentiment polarity of the target word. The objective of the CSCV model is to maximize Equation (4) where $\mathrm{S}_{\mathrm{w_t}}$ represents the sentiment polarity of the target word given its context.

$$\frac{1}{SC} \sum_{1}^{SC} \log p(S_{w_t} | w_{t-c}, \ldots, w_{t+c}) \tag{4}$$

3.4 Model Training

To train the CSCV model, we used the IMDB dataset [36] text data. It is a free and well-known dataset for sentiment analysis. It includes 50,000 reviews about movies with their sentiment labels (positive or negative). We constructed the pairs of «context window, target sentiment» the text provided by the IMDB dataset as described in Section 3.2. We have chosen this dataset because we planned to evaluate our models on datasets that contain movie reviews. This will help to partially alleviate the OOV (out of vocabulary) problem during the evaluation. Figure 4 shows the most relevant words after dataset pre-processing.

First, we normalized the datasets by converting all uppercases to lowercases and all digits to "D", fixing the contractions, and removing HTML tags in addition to abnormal sentences. In order to balance the word occurrences in the text, we performed subsampling of frequent words using the following sampling rate:

$$P(w_i) = \frac{10^{-3}}{p_i} (\sqrt{10^3 p_i + 1}) \tag{5}$$

Figure 4 Word Cloud of relevant words in the IMDB dataset.

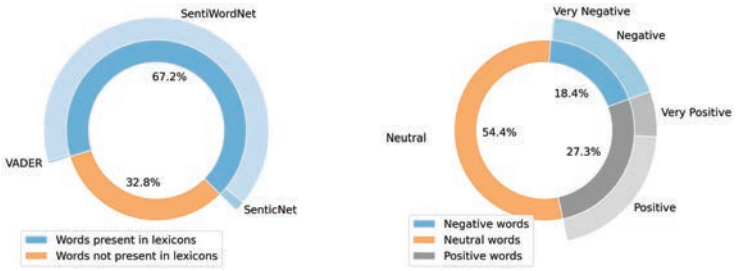

Figure 5 Summary statistics of words polarities and existence in the three lexicons.

where $P(w_i)$ represents the probability of keeping the word i, and p_i denotes the proportion of this word in the corpus. Therefore, the training text contains about 8 million words. The vocabulary size is about 62,000 with 67.2% of words existing in the combined lexicons. Figure 5 shows summary statistics about the polarity of words and their existence in the lexicons. As mentioned before, words that are not present in the lexicons are considered neutral. This explains the considerable number of neutral words in the vocabulary.

We learn the sentiment vectors by minimizing the loss function presented in Section 3.3. The embedding matrix $\mathrm{W} \in \mathbb{R}^{V \times N}$ is initialized randomly with values falling into $[-1, 1]$. Each row in W represents a vector of a unique word in the vocabulary. We empirically set the context window size as 2 and the embedding dimension as 300. This dimension has been selected for three reasons: (1) to help the model to capture some semantic and syntactic features in addition to sentiment orientation, (2) to ensure the stability of word embeddings, and (3) to facilitate the comparison with existing models. The training errors are optimized using the RMSprop optimization algorithm where the gradients are obtained via back-propagation [44].

The sentiment embeddings are extracted from the lookup table after the model ends training. The obtained vectors can be combined then with any pre-trained vectors applying PCA (Principal Component Analysis) technique:

$$EVs = \mathrm{PCA}(\mathrm{CSCV} \oplus \mathrm{PV}) \tag{6}$$

where, EVs, CSCV, PV refer to the result vectors called Enhanced Vectors, the sentiment embeddings obtained from our proposed model, and context-based pre-trained vectors, respectively. PCA aims to reduce the dimensionality of large data without removing important information. It is applied to keep sentiment information from CSCV vectors and semantic information from pre-trained vectors.

The performance of word embeddings is often evaluated using NLP tasks. Thus, we conducted experiments on sentiment classification using three DL models. Furthermore, we compared our refining method with the most popular word embeddings available online, including word2vec and GloVe. However, this method can be applied to any pre-trained model.

4 Experiments

This section is dedicated to describing the evaluation of our proposed approach. We used sentence-level sentiment analysis with deep learning to test the efficiency of the Enhanced Vectors. Thus, we carried out several experiments using well-known deep learning models and benchmarking datasets. However, the experimental settings are as follows:

Datasets: In this study, we used two well-known datasets to evaluate our approach. These datasets are Movie Review[1] (MR) [45], and Stanford Sentiments Treebank[2] (SST) [46]. The first dataset contains sentences describing movies (good or not good). These sentences were extracted from the Rotten Tomatoes website. If the sentence was marked "fresh" on the website it represents a positive review and vice versa. The SST dataset contains also sentences related to movie reviews and it has two variations: SST-1 and SST-2. The difference between them is that SST-1 is a fine-grained dataset that involves five labels. SST-2 includes only the existing positive and negative reviews in SST-1. In Table 1, we summarize the statistics of all datasets after pre-processing. CV (cross-validation) means the dataset does not have

[1] http://www.cs.cornell.edu/home/llee/papers/pang-lee-stars.home.html
[2] http://nlp.stanford.edu/sentiment/

Table 1 Summary statistics for the datasets used for evaluation. C, SL, and VS denote the number of classes, sentence length, and vocabulary size respectively

Dataset	C	Max SL	VS	Size	Train	Dev	Test
MR	2	56	18765	10662	9596	–	CV
SST-1	5	53	17836	11855	8544	1101	2210
SST-2	2	53	16188	9613	6920	872	1821

Table 2 Hyperparameters selected for each classifier

Hyperparameter	CNN	LSTM	BiLSTM
Filter Number	128	–	–
Filter Size	3, 4, 5	–	–
Pooling Size	Sequence Length – Filter Size + 1	–	–
LSTM Dim	–	164	164
Dropout	0.5	–	–
Regularization	L2(0.01)	L2(0.01)	L2(0.01)
Epoch	40	20	20
Batch Size	32	32	32
Optimizer	Adam	Adam	Adam

a standard train, dev, and test split, therefore 10-fold CV was used to test the performance of the trained classifier. To evaluate the performance of a sentiment classification model, these datasets have been widely used as a benchmark in many studies. Thus, in this research, the experiments are conducted with two types of classification: binary and multi-class, to compare our proposed approach with the baseline methods.

Word embedding: The pre-trained word embeddings combined with our sentiment embeddings are word2vec [9, 10] and Glove [11]. Both pre-trained word embeddings are available online. Word2vec was trained on the GoogleNews dataset while GloVe was trained on Common Crawl 42B.

Classifiers: In this study we used the following classifiers: Convolutional Neural Network (CNN) [47], Long Short Term Memory (LSTM) [48] and Bidirectional LSTM (BiLSTM) [49]. For CNN we follow [47]'s work by using the static method with default settings for both datasets. The two other models are implemented following the methods proposed in [49]. We have implemented these classifiers using TensorFlow and Keras with the default hyperparameters provided by the used libraries and the original papers. The best models with convenient hyperparameters are selected based on the accuracy obtained for the development set. Table 2 summarizes the hyperparameters chosen for each classifier.

Evaluation metric: Since the datasets have a balanced number of samples of all classes, we adopt the accuracy metric to evaluate the performance of our approach. The accuracy represents the ratio between the correctly predicted examples to the total number of examples, which is defined as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \in [0, 1] \tag{7}$$

Baseline method: The following methods are used as benchmarks for sentence-level sentiment classification:

- **CNN-non-static**: 1d-CNN with pre-trained word embeddings as inputs proposed by [47].
- **CNN-multichannel**: 1d-CNN but with two sets of word embeddings proposed by [47].
- **Tree LSTM**: A generalization of LSTMs to tree-structured network topologies proposed by [49].
- **Multi-task LSTM**: A multi-task learning framework to jointly learn across multiple related tasks proposed by [50].
- **BiLSTM-CRF**: A two-step pipeline framework for sentence-level sentiment classification proposed by [48]
- **BiLSTM+Re(word2vec)**: BiLSTM neural network with word2vec refined using intensity score proposed by [25].
- **BiLSTM+Re(GloVe)**: BiLSTM neural network with GloVe refined using intensity score proposed by [25].
- **CNN-non-static + IWV(WG):** 3d-CNN with Improved Word Vector (IWV) for word2vec and GloVe (WG) proposed by [24].
- **CNN-multichannel + IWV(WG):** 3d-CNN with Improved Word Vector (IWV) for word2vec and GloVe (WG) using two sets of word embeddings proposed by [24].
- **AC-BiLSTM**: Attention-based bidirectional long short-term memory with convolution layer proposed by [51].
- **BiLSTM+SAWE-word2vec(PCA100)**: BiLSTM neural network with sentiment-aware word embeddings combined with word2vec proposed by [23].
- **BiLSTM+SAWE-GloVe(PCA30)**: BiLSTM neural network with sentiment-aware word embeddings combined with GloVe proposed by [23].

5 Evaluation

5.1 Qualitative Evaluation

In order to verify the effect of sentiment contextualized vectors, we manually check the pre-trained word embeddings after refinement. 10 nearest neighbors of several words derived from word2vec and GloVe are selected for evaluation as shown in Figure 6, and Figure 7. The criterion of selecting examples is whether the word has a positive or negative polarity to demonstrate if the refinement method can remove the noisy words (words with opposite sentiment polarity in the red cells) from the top similar words. Based on word embeddings, the nearest neighbors are obtained using cosine similarities between the selected words and other words in the vocabulary.

As indicated in Figure 6, the nearest neighbors derived from word2vec are not so good. word2vec has at least one word with opposite sentiment polarity in the top nearest neighbors for each example. This is because word2vec depends totally on the local context to predict words. For GloVe, the problem of opposite polarity words is not often existed. However, the nearest words generated from GloVe embeddings may contain some words that could not be considered similar (e.g., "because" and "too") but they are often used in the same context. The reason for that is Glove does not rely only on the local context but incorporates words co-occurrences (global statistics) to generate word embeddings. To the best of our knowledge, such words do not affect much the performance of sentiment analysis because they are considered neutral.

word2vec

Word	Polarity	Score
good	SP	0.56
great	P	0.25
bad	N	-0.59
terrific	P	0.04
decent	P	0.39
nice	P	0.47
excellent	SP	1.00
fantastic	P	0.10
better	SP	0.52
solid	P	0.03
lousy	SN	-0.70

PCA(word2vec and CSCV)

Word	Polarity	Score
good	SP	0.56
nice	P	0.47
terrific	P	0.04
fantastic	P	0.10
decent	P	0.39
perfect	P	0.37
fabulous	P	0.16
wise	P	0.16
awesome	SP	0.75
marvelous	P	0.25
unbelievable	SP	0.50

word2vec

Word	Polarity	Score
bad	SN	-0.59
good	SP	0.56
terrible	SN	-0.62
horrible	SN	-0.62
Bad	SN	-0.60
lousy	SN	-0.70
crummy	SN	-0.75
horrid	SN	-0.88
awful	N	-0.21
dreadful	SN	-0.70
horrendous	SN	-0.62

PCA(word2vec and CSCV)

Word	Polarity	Score
bad	SN	-0.59
lousy	SN	-0.70
terrible	SN	-0.62
horrible	SN	-0.62
nasty	SN	-0.78
crummy	SN	-0.75
worse	SN	-0.54
dumb	N	-0.22
cruddy	SN	-0.62
stupid	SN	-0.34
scary	SN	-0.75

Figure 6 Example of nearest neighbors of words "good" and "bad" from word2vec and its refined word embeddings. SP, P, N, and SN denote strong positive, positive, negative, and strong negative, respectively.

GloVe

Word	Polarity	Score
gorgeous	SP	0.75
beautiful	SP	0.68
lovely	P	0.46
stunning	P	0.12
fabulous	P	0.16
amazing	P	0.15
wonderful	SP	0.75
adorable	SP	0.5
exquisite	P	0.31
fantastic	P	0.10
delightful	SP	0.75

PCA(GloVe and CSCV)

Word	Polarity	Score
gorgeous	SP	0.75
beautiful	SP	0.68
stunning	P	0.12
fabulous	P	0.16
amazing	P	0.15
cute	SP	0.56
exquisite	P	0.31
elegant	P	0.38
nice	P	0.46
beautifully	P	0.38
pretty	P	0.08

GloVe

Word	Polarity	Score
unhappy	SN	-0.69
frustrated	N	-0.21
dissatisfied	N	-0.19
disappointed	N	-0.38
miserable	SN	-0.69
anxious	N	-0.19
worried	N	-0.20
annoyed	N	-0.38
disgruntled	SN	-0.56
understandably	P	0.13
unfortunate	SN	-0.73

PCA(GloVe and CSCV)

Word	Polarity	Score
unhappy	SN	-0.69
frustrated	N	-0.21
dissatisfied	N	-0.19
disappointed	N	-0.38
miserable	SN	-0.69
angry	N	-0.33
sad	SN	-0.63
fearful	N	-0.48
tired	N	-0.38
upset	N	-0.24
depressed	N	-0.19

Figure 7 Example of nearest neighbors of words “gorgeous” and “unhappy” from GloVE and its refined word embeddings. SP, P, N, and SN denote strong positive, positive, negative, and strong negative, respectively.

Even the refining method changes the order of the nearest neighbors in the examples without opposite polarity words, it does not affect the quality of word embeddings as indicated in Figure 7. Some words can be replaced, but with other sentimentally similar words. Nevertheless, it can remove noisy words as in the second example. Based on the manual check this approach tends to give better results with word2vec because they follow the same principle. However, the generated word embeddings contain sentimental information which they are very important for SA and can be used with any other pre-trained word embeddings. These results indicate that our approach can improve the quality of a pre-trained word vector, so it can be closer to both semantically and sentimentally similar words.

5.2 Sentiment Classification

Sentiment classification is the process of classifying given instances into coarse-grained classes such as positive, negative, and neutral, or fine-grained classes using machine learning algorithms [52]. Figure 8, Figure 9, and Figure 10 illustrate the results of the benchmark datasets. Figure 8 shows the performance obtained from EVs with word2vec pre-trained word embeddings for all datasets. As can be seen for MR, our approach outperformed word2vec in most cases. The accuracy of the CNN and BiLSTM models has improved by 1.3% and 0.3%, respectively. Similarly, EVs provide absolute accuracy improvement for SST, especially for fine-grained. The accuracy increased by 0.2%, 1.6%, and 2.3% for CNN, LSTM, and BiLSTM successively.

Figure 9, indicates the comparison between GloVe and its refined word embeddings over all datasets. As can be seen, EVs give an improvement of

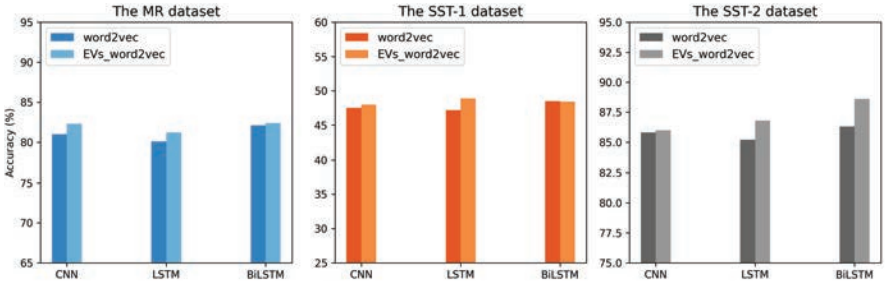

Figure 8 The accuracy comparison of word2vec and refined word embeddings over MR, SST-1, and SST-2 using CNN, LSTM, and BiLSTM models.

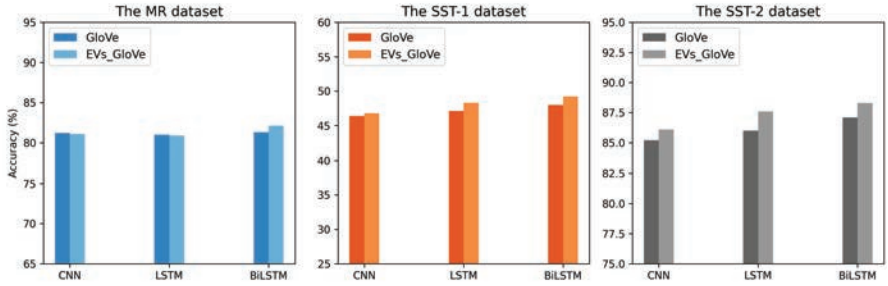

Figure 9 The accuracy comparison of GloVe and refined word embeddings over MR, SST-1, and SST-2 using CNN, LSTM, and BiLSTM models.

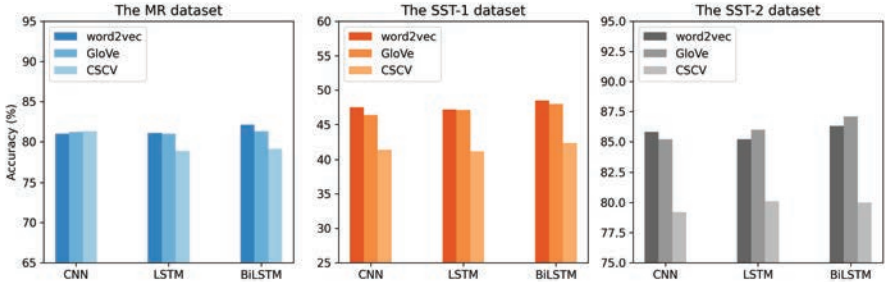

Figure 10 The accuracy comparison of GloVe, word2vec, and CSCV over MR, SST-1, and SST-2 using CNN, LSTM, and BiLSTM models.

0.4%, 1.2%, and 1.2% for CNN, LSTM, and BiLSTM models respectively on the SST-1 dataset. The same situation can be observed in the binary task with an improvement of accuracy in the three classification models, especially for LSTM (accuracy is improved by 1.6%). For MR, GloVe performs better with CNN and LSTM. However, the results are closer for both word embeddings.

Table 3 State-of-the-art results on the MR, SST-1, and SST-2 datasets. The best results are highlighted in boldface

Method	MR	SST-1	SST-2
CNN-non-static	81.5	48.0	87.2
CNN-multichannel	81.1	47.4	88.1
Tree LSTM	–	50.6	86.9
Multi-task LSTM	–	49.6	87.9
BiLSTM-CRF	82.3	48.5	88.3
BiLSTM+Re(word2vec)	–	49.6	88.2
BiLSTM+Re(GloVe)	–	49.7	88.6
CNN-non-static + IWV(WG)	82.0	47.0	86.5
CNN-multichannel + IWV(WG)	81.5	46.4	87.0
AC-BiLSTM	–	48.9	88.3
BiLSTM+SAWE-word2vec(PCA100)	–	49.8	87.2
BiLSTM+SAWE-GloVe(PCA30)	–	**52.1**	88.5
BiLSTM + EV_Glove	82.1	49.2	88.3
BiLSTM + EV_word2vec	**82.4**	48.4	**88.6**

We also evaluated our sentiment contextualized embedding by using them directly with the classification models. This approach shows promising results as indicated in Figure 10. Although word2vec and GloVe perform better in most cases, CSCV has achieved state-of-the-art even if it does not contain much semantic and syntactic information. As observed in Figure 10, it outperforms word2vec and GloVe on the MR dataset using CNN with 0.2% and 0.1%, respectively.

To evaluate the performance of our approach we compared our results with ones obtained by 11 state-of-the-art methods. Table 3, indicates the results of these methods including the one used in this study. However, our approach achieved promising results on the MR and SST-2 datasets. The selected methods are a collection of approaches that use generic algorithms such as CNN [47] and AC-BiLSTM [51], or they leverage the integration of word sentiment knowledge into DL models.

As shown in Table 3, the BiLSTM model with refined word2vec achieves the best results. Considering its accuracy of 82.4% on the MR dataset, it outperforms BiLSTM-CRF [48] and CNN with IWV [24]. The same model performs well on the SST-2 dataset and achieved an accuracy of 88.6%. With this result, the model outperforms most of the complex methods such as Multitask LSTM [50], Tree LSTM [49], and AC-BiLSTM [51]. For the SST-1 dataset, BiLSTM with SAWE-GloVe(PCA30) has achieved the best

accuracy of 52.1%. However, the refined GloVe with CSCV reaches a good performance compared to BiLSTM-CRF [48] and AC-BiLSTM [51].

6 Discussion

As illustrated in Figures 6 and 7, the enhanced vectors removed semantically similar but sentimentally dissimilar nearest neighbor words from the top 10 similar words. Therefore, our approach provides high-quality word embedding for sentiment analysis. The vectors obtained from this approach can be combined with any other context-based word embedding to avoid the problem of closest words with opposite sentiment polarity. This will help to improve the performance of sentiment analysis. The results show that our refining method achieved a good accuracy than some complex models and reached the state-of-the-art applying BiLSTM technique. This is because the enhanced vectors contain the important information for sentiment analysis which are the words' sentiment orientations.

As can be observed from qualitative evaluation and sentiment classification, the proposed approach tends to give more accurate results with the refined word embeddings of word2vec. This is because CSCV follows the same principle as the CBOW model. However, with the two pre-trained word embeddings used in the experiments, our refinement approach was able to outperform the non-static methods such as Tree LSTM and CNN-non-static. These methods realize their best results using trainable embeddings, while the experiments in this study are conducted with static refined embeddings. This demonstrates the effectiveness of CSCV sentiment embeddings and the refined approach. The use of CSCV sentiment embeddings directly attains good results, but not as other embeddings. This is because the CSCV model does not capture many semantic and syntactic features of words. The selected 300-dimension embedding size is to make the model encode the full sentimental orientation of words in addition to some semantic and syntactic information.

7 Conclusion and Future Works

In this study, we introduced a novel approach to creating sentiment embeddings from local contexts and combining them with other pre-trained word embeddings. The first step of this approach follows the precept of the CBOW model. It aims to predict the sentiment of a center word from surrounding

context words. The word's sentiment is acquired from three well-known lexicons, namely: SentiWordNet, SenticNet, and VADER. The second step of our approach combines the obtained sentiment embeddings with other pre-trained word embeddings applying the PCA technique. The combination seeks to retain semantic information from pre-trained word embedding and sentiment information from the CSCV model. Therefore, providing high-quality word embeddings for many NLP tasks, especially text-based SA and emotion detection. The experiments on different benchmark datasets show that our refining approach improves significantly the performance of the sentiment analysis task. Moreover, it outperforms state-of-the-art methods over MR and SST-2 datasets. This reveals how our approach is effective in encoding words' sentimental information into word embeddings. In the future, we will extend the proposed approach by trying deeper architectures and efficient methods of encoding sentiment information. Since the quality of word embeddings depends on a certain number of features, encoding additional information such as POS tags can also be a part of our future work. However, as this approach relies on sentiment lexicons the most important work in the future is to build a large and accurate one. Applying our approach in some user-centric situations will raise privacy concerns; thus, we plan on addressing this problem in the future using encryption-based techniques such as homomorphic encryption, which showed promising results in previous works [53–55].

References

[1] Birjali, M., Kasri, M., and Beni-Hssane, A., 2021, "A Comprehensive Survey on Sentiment Analysis: Approaches, Challenges and Trends," Knowledge-Based Syst., pp. 1–26.

[2] El-Ansari, A., Beni-Hssane, A., and Saadi, M., 2020, "An Improved Modeling Method for Profile-Based Personalized Search," *Proceedings of the 3rd International Conference on Networking, Information Systems & Security*, ACM, New York, NY, USA, pp. 1–6.

[3] El-Ansari, A., Beni-Hssane, A., and Saadi, M., 2017, "A Multiple Ontologies Based System for Answering Natural Language Questions," pp. 177–186.

[4] Atzeni, M., and Reforgiato Recupero, D., 2020, "Multi-Domain Sentiment Analysis with Mimicked and Polarized Word Embeddings for Human–Robot Interaction," Futur. Gener. Comput. Syst., **110**, pp. 984–999.

[5] Ali, F., Kwak, D., Khan, P., El-Sappagh, S., Ali, A., Ullah, S., Kim, K. H., and Kwak, K.-S., 2019, "Transportation Sentiment Analysis Using Word Embedding and Ontology-Based Topic Modeling," Knowledge-Based Syst., **174**, pp. 27–42.

[6] Dessí, D., Dragoni, M., Fenu, G., Marras, M., and Reforgiato Recupero, D., 2020, "Deep Learning Adaptation with Word Embeddings for Sentiment Analysis on Online Course Reviews," pp. 57–83.

[7] Kaibi, I., Nfaoui, E. H., and Satori, H., 2020, "Sentiment Analysis Approach Based on Combination of Word Embedding Techniques," pp. 805–813.

[8] Kasri, M., Birjali, M., and Beni-Hssane, A., 2019, "A Comparison of Features Extraction Methods for Arabic Sentiment Analysis," *Proceedings of the 4th International Conference on Big Data and Internet of Things*, ACM, New York, NY, USA, pp. 1–6.

[9] Mikolov, T., Chen, K., Corrado, G., and Dean, J., 2013, "Efficient Estimation of Word Representations in Vector Space," pp. 1–12.

[10] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J., 2013, "Distributed Representations of Words and Phrases and Their Compositionality."

[11] Pennington, J., Socher, R., and Manning, C. D., 2014, "GloVe: Global Vectors for Word Representation," *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.

[12] Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T., 2016, "Enriching Word Vectors with Subword Information," pp. 1–12.

[13] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L., 2018, "Deep Contextualized Word Representations."

[14] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K., 2018, "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding."

[15] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I., 2019, "Language Models Are Unsupervised Multitask Learners."

[16] Araque, O., Corcuera-Platas, I., Sánchez-Rada, J. F., and Iglesias, C. A., 2017, "Enhancing Deep Learning Sentiment Analysis with Ensemble Techniques in Social Applications," Expert Syst. Appl., **77**, pp. 236–246.

[17] Giatsoglou, M., Vozalis, M. G., Diamantaras, K., Vakali, A., Sarigiannidis, G., and Chatzisavvas, K. C., 2017, "Sentiment Analysis Leveraging Emotions and Word Embeddings," Expert Syst. Appl., **69**, pp. 214–224.

[18] Kasri, M., Birjali, M., and Beni-Hssane, A., 2021, "Word2Sent: A New Learning Sentiment-Embedding Model with Low Dimension for Sentence Level Sentiment Classification," Concurr. Comput. , **33**(9), pp. 1–12.

[19] Mikolov, T., Yih, W., and Zweig, G., 2013, "Linguistic Regularities in Continuous Space Word Representations," *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751.

[20] Erritali, M., Beni-Hssane, A., Birjali, M., and Madani, Y., 2016, "An Approach of Semantic Similarity Measure between Documents Based on Big Data," Int. J. Electr. Comput. Eng., **6**(5), pp. 1–10.

[21] El-Ansari, A., Beni-Hssane, A., Saadi, M., and El Fissaoui, M., 2021, "PAPIR: Privacy-Aware Personalized Information Retrieval," J. Ambient Intell. Humaniz. Comput., **12**(10), pp. 9891–9907.

[22] El-Ansari, A., Beni-hssane, A., and Saadi, M., 2020, "An Ontology-Based Profiling Method for Accurate Web Personalization Systems."

[23] Naderalvojoud, B., and Sezer, E. A., 2020, "Sentiment Aware Word Embeddings Using Refinement and Senti-Contextualized Learning Approach," Neurocomputing, **405**, pp. 149–160.

[24] Rezaeinia, S. M., Rahmani, R., Ghodsi, A., and Veisi, H., 2019, "Sentiment Analysis Based on Improved Pre-Trained Word Embeddings," Expert Syst. Appl., **117**, pp. 139–147.

[25] Yu, L.-C., Wang, J., Lai, K. R., and Zhang, X., 2018, "Refining Word Embeddings Using Intensity Scores for Sentiment Analysis," IEEE/ACM Trans. Audio, Speech, Lang. Process., **26**(3), pp. 671–681.

[26] Baccianella, S., Esuli, A., and Sebastiani, F., 2010, "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining," *Proceedings of the International Conference on Language Resources and Evaluation, {LREC} 2010, 17–23 May 2010, Valletta, Malta*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner, and D. Tapias, eds., European Language Resources Association.

[27] Cambria, E., Poria, S., Hazarika, D., and Kwok, K., 2018, "SenticNet 5: Discovering Conceptual Primitives for Sentiment Analysis by Means of Context Embeddings," *Thirty-Second AAAI Conference on Artificial Intelligence*.

[28] Hutto, C. J., and Gilbert, E., 2015, "VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text," *Proceedings of the 8th International Conference on Weblogs and Social Media, ICWSM 2014*.

[29] Kasri, M., Birjali, M., El-Ansari, A., and Beni-Hssane, A., 2021, "Enhanced Word Embeddings with Sentiment Contextualized Vectors for Sentiment Analysis," *The International Conference on Information, Communication & Cybersecurity, ICI2C'21*, Khouribga, Morocco, pp. 1–10.

[30] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C., 2003, "A Neural Probabilistic Language Model," J. Mach. Learn. Res., **3**(null), pp. 1137–1155.

[31] Collobert, R., and Weston, J., 2008, "A Unified Architecture for Natural Language Processing," *Proceedings of the 25th International Conference on Machine Learning – ICML '08*, ACM Press, New York, New York, USA, pp. 160–167.

[32] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P., 2011, "Natural Language Processing (Almost) from Scratch."

[33] Levy, O., and Goldberg, Y., 2014, "Dependency-Based Word Embeddings," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Association for Computational Linguistics, Baltimore, Maryland, pp. 302–308.

[34] Radford, A., 2018, "Improving Language Understanding by Generative Pre-Training."

[35] Hu, B., Tang, B., Chen, Q., and Kang, L., 2016, "A Novel Word Embedding Learning Model Using the Dissociation between Nouns and Verbs," Neurocomputing, **171**, pp. 1108–1117.

[36] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C., 2011, "Learning Word Vectors for Sentiment Analysis," *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, Portland, Oregon, USA, pp. 142–150.

[37] Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., and Qin, B., 2014, "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification," *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 1555–1565.

[38] Ren, Y., Zhang, Y., Zhang, M., and Ji, D., 2016, "Improving Twitter Sentiment Classification Using Topic-Enriched Multi-Prototype Word

Embeddings," *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI Press, pp. 3038–3044.

[39] Lan, M., Zhang, Z., Lu, Y., and Wu, J., 2016, "Three Convolutional Neural Network-Based Models for Learning Sentiment Word Vectors towards Sentiment Analysis," *2016 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 3172–3179.

[40] Warriner, A. B., Kuperman, V., and Brysbaert, M., 2013, "Norms of Valence, Arousal, and Dominance for 13,915 English Lemmas," Behav. Res. Methods, **45**(4), pp. 1191–1207.

[41] Yin, R., Li, P., and Wang, B., 2017, "Sentiment Lexical-Augmented Convolutional Neural Networks for Sentiment Analysis," *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*, IEEE, pp. 630–635.

[42] Yu, L.-C., Wang, J., Lai, K. R., and Zhang, X., 2017, "Refining Word Embeddings for Sentiment Analysis," *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Stroudsburg, PA, USA, pp. 534–539.

[43] Miller, G. A., 1995, "WordNet: A Lexical Database for English," Commun. ACM, **38**(11), pp. 39–41.

[44] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., 1986, "Learning Representations by Back-Propagating Errors," Nature, **323**(6088), pp. 533–536.

[45] Pang, B., and Lee, L., 2005, "Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales."

[46] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C., 2013, "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank," *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Seattle, Washington, USA, pp. 1631–1642.

[47] Kim, Y., 2014, "Convolutional Neural Networks for Sentence Classification."

[48] Chen, T., Xu, R., He, Y., and Wang, X., 2017, "Improving Sentiment Analysis via Sentence Type Classification Using BiLSTM-CRF and CNN," Expert Syst. Appl., **72**, pp. 221–230.

[49] Tai, K. S., Socher, R., and Manning, C. D., 2015, "Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks," *Proceedings of the 53rd Annual Meeting of the Association for*

Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Association for Computational Linguistics, Beijing, China, pp. 1556–1566.

[50] Liu, P., Qiu, X., and Huang, X., 2016, "Recurrent Neural Network for Text Classification with Multi-Task Learning."

[51] Liu, G., and Guo, J., 2019, "Bidirectional LSTM with Attention Mechanism and Convolutional Layer for Text Classification," Neurocomputing, **337**, pp. 325–338.

[52] Chen, X., Rao, Y., Xie, H., Wang, F. L., Zhao, Y., and Yin, J., 2019, "Sentiment Classification Using Negative and Intensive Sentiment Supplement Information," Data Sci. Eng., **4**(2), pp. 109–118.

[53] El Makkaoui, K., Ezzati, A., Beni-Hssane, A., and Motamed, C., 2016, "Cloud Security and Privacy Model for Providing Secure Cloud Services," *2016 2nd International Conference on Cloud Computing Technologies and Applications (CloudTech)*, IEEE, pp. 81–86.

[54] El Makkaoui, K., Beni-Hssane, A., and Ezzati, A., 2019, "Speedy Cloud-RSA Homomorphic Scheme for Preserving Data Confidentiality in Cloud Computing," J. Ambient Intell. Humaniz. Comput., **10**(12), pp. 4629–4640.

[55] El Makkaoui, K., Ezzati, A., and Beni-Hssane, A., 2017, "Cloud-RSA: An Enhanced Homomorphic Encryption Scheme," pp. 471–480.

Biographies

Mohammed Kasri is a PhD student received the bachelor's degree in computer science from Ibn Tofail University in 2011 and the master's degree in computer science from Chouib Doukkali University in 2014. Fields of interest: Sentiment Analysis, Machine Learning, Big Data and Programming Languages.

Marouane Birjali received his PhD. degree in computer science from the Faculty of Sciences, Chouaïb Doukkali University, El Jadida since 2019. Currently, he is a Researcher in the same faculty and working as an IT engineer. His research interests include Big Data, AI and Sentiment Analysis.

Mohamed Nabil received the B.Sc. degree in Computer Sciences from Hassan 1st University, Faculty of Sciences and Technical of Settat in Morocco, in 2001, and a M.Sc. degree in engineering decision from the Hassan 1st University, Faculty of Sciences and Techniques of Settat in Morocco, in 2008. Professor of Computer Sciences in high school – from 2002 to 2019. Assistant Professor at the Faculty of Sciences of El-jadida in Morocco – since 2019. H's a Member of LaROSERI at Faculty of Sciences of El-jadida. His current research interests are: Vehicular Ad hoc Networks (Security and QoS), Simulation Network Performance, Network Protocols and Analysis of Quality of Service in Next Generation Networks, Natural Language Processing, and Game Theory.

Abderrahim Beni-Hssane received his Ph.D. degree in computer science from Mohamed V University, Rabat, Morocco, in 1997. Since September 1994, he has been a Researcher and a Professor at the Science Faculty, Chouaib Doukkali University, El Jadida, Morocco. His research interests include Performance evaluation in wireless networks, Cryptography, Sentiment Analysis, Cloud Computing, and Big Data.

Anas El-Ansari received his PhD. degree in computer science from the Faculty of Sciences, Chouaïb Doukkali University, El Jadida. Currently, he is a Researcher and a Professor with Polydisciplinary Faculty of Nador, Mohamed First University, Morocco. His research interests include Recommender systems, Cryptography, Privacy, Sentiment Analysis and Semantic Web.

Mohamed El Fissaoui is a researcher and professor at High School of Technology of Nador, Mohammed First University Oujda, Morocco. He received a master degree in computer science and a Ph.D degree in Computer Sciences from the faculty of sciences, Chouaîb Doukkali University, El Jadida, Morocco. His current interests include developing a specification and design techniques for use within Intelligent Network, data mining, Big data, information Retrieval, Mobile Agents, Vanets. He is also a member of MASI laboratory, at FPN, Nador.