Received: 7 February 2018 | Revised: 5 May 2018 | Accepted: 9 May 2018
DOI: 10.1002/cpe.4783

WILEY

SPECIAL ISSUE PAPER

Evaluating deep learning models for sentiment classification

Betül Ay Karakuş[1] | Muhammed Talo[2] | İbrahim Rıza Hallaç[1] | Galip Aydin[1]

[1]Department of Computer Engineering, Fırat University, Elazığ,Turkey
[2]Department of Computer Engineering, Munzur University, Tunceli,Turkey

Correspondence
Betül Ay Karakuş, Department of Computer Engineering, Fırat University, 23119 Elazığ, Turkey.
Email: betulay@firat.edu.tr

Summary

Deep learning has emerged as an effective solution to various text mining problems such as document classification and clustering, document summarization, web mining, and sentiment analysis. In this paper, we describe our work on investigating several deep learning models for a binary sentiment classification problem. We used movie reviews in Turkish from the website www.beyazperde.com to train and test the deep learning models. We also report a detailed comparison of the models in terms of accuracy and time performances. Two major deep learning architectures used in this study are Convolutional Neural Networks and Long Short-Term Memory. We built several variants of these models by changing the number of layers, tuning the hyper-parameters, and combining models. Additionally, word embeddings were created by applying the word2vec algorithm with a skip-gram model on a large dataset (∼ 13 M words) composed of movie reviews. We investigate the effect of using the pre-word embeddings with these models. Experimental results have shown that the use of word embeddings with deep neural networks effectively yields performance improvements in terms of run time and accuracy.

KEYWORDS

big data, CNN, deep learning, LSTM, sentiment classification, word embeddings

1 | INTRODUCTION

Extracting sentiments from textual data preserves its popularity among natural language processing (NLP) tasks. Together with the availability of online large labeled text data and the rise of deep learning (DL), methods in the machine learning (ML) field are leading NLP applications to be revisited by the researchers. Identifying the sentiment of individuals toward specific events or products enables decision makers to plan relevant tasks in a more opinion-aware manner.

There are different resources for gathering information that contain opinions such as emails, online questionnaires, social media posts, and blog posts. The amount of data collected from these resources is usually too big for manual analyses by humans. On the other hand, they are too valuable to be ignored. Sentiment analysis tries to solve this problem by training the model on already identified or tagged opinions with different polarities and by using this model to predict unidentified or untagged opinions.

Cui et al[1] presented their experiments on sentiment analysis of online brand reviews. They discussed the aspects of techniques in 3 groups as Passive-Aggressive (PA) Algorithm Based Classifier, Language Modeling (LM) Based Classifier, and Winnow Classifier.[2] For feature selection, they addressed the role of high order n-grams as linguistic features. Socher et al[3] proposed a new model called Recursive Neural Tensor Network for identifying sentences as positive or negative by using fully labeled parse trees. They use a dataset based on a corpus of 11,855 movie reviews.[4]

Traditional supervised and unsupervised ML algorithms such as Support Vector Machines (SVM), Naive Bayes, K Nearest Neighbor, and Maximum Entropy have widely been used in sentiment analysis. However, deep learning architectures such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM) Networks have been chosen more frequently recently. Since no universal DL model exists for all type of problems, an appropriate model should be selected according to the problem domain. Different network models give varying performance results on different application domains such as computer vision and information retrieval.[5]

Sarkar et al[6] stated that recent approaches for sentiment analysis have used text features based on word embeddings like GloVe[7] and word2vec.[8] Previous studies commonly used other important text representation models such as Bag of Words or TF-IDF. However, in recent years, more recent word embedding approaches such as word2vec are being utilized in sentiment analysis studies.[9] In this study, we also use the word2vec model. Word embeddings were generated using our dataset of Turkish movie reviews, which consists of around 13 million words.

The rest of the paper is organized as follows. Section 2 presents a literature review of sentiment analysis with different machine learning methods and also points out important developments in the representation of textual data, which is closely associated with sentiment analysis tasks. In Section 3, we introduce various deep learning models used in this study. Section 4 outlines our experiments on sentiment classification of movie reviews with different deep learning models. The results of the experiments are shared in Section 5.

2 | BACKGROUND

Sentiment analysis, also known as opinion mining, is one of the problems of NLP and text mining.[10] It is an important area that is being studied both in academia and in various commercial applications. Sentiment analysis, which can also be described as a classification task, aims to predict the general emotion of the text that might be a tweet or a review about movies, music, or products. The major goal is to find out whether the opinion expressed in the text is positive or negative, in some cases with a score or confidence metric.

Recent studies on sentiment analysis have been summarized in the works of Pang and Lee,[11] Catal and Nangir,[12] and Agarwal and Mittal.[13] SVM, Naive Bayes, KNN, and Maximum Entropy algorithms are widely used in sentiment analysis studies on movie reviews. Kalaivani and Shunmuganathan[14] performed emotion classification by applying unsupervised machine learning algorithms on a dataset containing 1000 positive and 1000 negative reviews. The study reports that SVM performed better classification with 80% accuracy compared with the Naive Bayes and KNN algorithms. Another study[15] reported that the Naive Bayes Algorithm achieved a better classification result than the SVM with an accuracy of 65% on 1400 positive and negative comments from the IMDB (imdb.com) movie reviews database. Bhadane et al[16] used SVM to add certain static properties to classify a given text as positive or negative. 78% accuracy has been achieved on the comments that have been pre-processed such as removing common words (stopwords), finding the roots of words, removing words that repeat more than 30 and less than 4 times, and the use of WordNet that includes relationships such as synonymy and antagonism between words. In the sentiment analysis studies with traditional machine learning methods, the problem of dimensionality[17] and the change of accuracy rates depending on the feature selections are important problems. Certain learning algorithms behave badly and get inaccurate results when high-dimensional datasets are used.

Recent developments in machine learning made it possible to construct more complex models for processing and analyzing big data. The models developed for text analysis such as n-gram, bag of words, and word frequencies (TF-IDF) have clear superiority over simple models used in common methods. Distributed representation of the words is one of the most successful models developed recently.

Distributed representations of words[18] are obtained from various artificial neural network models and are generally superior to n-gram models.[19] Mikolov et al presented a new architecture for distributed word representations with a class of methods called Neural Language Models.[20] They developed a simple logarithmic linear classification network with word2vec software including skip-gram models.[8] The skip-gram model is an effective way to learn the word representations (embeddings) with high accuracy using a large amount of unstructured text data. Most of the deep learning methods used in natural language processing studies learn the representations of word obtained from neural language models.

Although neural network is not a new concept, in the past decade, it has gained popularity due to progresses made in the computing capacity and the development of more efficient models that can be trained on big data.

One of the major fields where deep learning achieves successful results is natural language processing (NLP). Ouyang et al[21] applied the deep neural network architecture he created using word2vec and CNN, on a dataset of 11855 comments with five different classes (positive, negative, slightly positive, slightly negative, and neutral). He observed that the model he created achieves better classification results with 45.4% accuracy when compared with Recurrent Neural Networks (RNN) and Matrix-Vector Recurrent Neural Networks (MV-RNN). In another study,[22] using LSTM and CNN neural network models, sentiment analysis was performed on user comments obtained from IMDB and Yelp (yelp.com/dataset_challenge) datasets.

Zhou[23] describes sentiment classification on movie reviews using the Stanford Sentiment Treebank (SST)[3] dataset. Architectures consisting of a combination of CNN and LSTM models have achieved better results than multilayered CNN and RNN models. An 87.7% accuracy is achieved for the 2-class (positive and negative) and a 49.2% accuracy for the 5-class (very positive, positive, neutral, very negative, negative) dataset. In another study,[23] sentiment analysis was performed for two classes on the SST dataset. A 3-structured LSTM architecture was presented that achieves 88% correct classification with GloVe vectors. Kim[24] performed sentiment classification on various datasets using a simple CNN model trained on word2vec. An accuracy of 93.2% on the Subjectivity dataset,[25] which was used to classify an object as objective or subjective; 85% on the CR dataset with positive and negative customer comments[26]; and 88.1% on the SST dataset containing positive and negative movie reviews were obtained. In the study, conducted by Collobert et al,[9] the use of word2vec embeddings trained on English Wikipedia has shown a significant increase in performance. It is unclear whether this is because of the architecture proposed by Mikolov et al[8] or of 100 million words trained in Google news.

Sentiment classification on the IMDB dataset with 50,000 reviews has been studied in the works of Wen et al[27] and Hong and Fang.[28] Wen et al[27] reported test accuracy results between 88% and 90% for various DL models. In the work of Hong and Fang,[28] for the LSTM model, 98.3% training accuracy is obtained and, for the 2-Layer MLP with Paragraph Vectors, 97.1% training accuracy is obtained.

Classification of the large-scale datasets has also been looked at in different studies.[29-31] However, their method focused on converting the dataset to a reduced data representation and then apply classification. There have been studies on various big data processing techniques used particularly for monitoring events.[32] However, our focus in this paper is conducting sentiment analysis on large-scale data.

Sentiment classification studies in the literature are usually focused on the English language. In this paper, sentiment classification was performed on Turkish movie reviews using pre-trained word embeddings and popular deep learning networks. Since no standard sentiment classification dataset exist in Turkish, we created a dataset with movie reviews that are collected from the website www.beyazperde.com, a movie database with user and editor reviews.

3 | DEEP LEARNING MODELS FOR SENTIMENT CLASSIFICATION

3.1 | Sentiment analysis

Sentiment analysis is a popular data mining and NLP topic both in the industry and academia.[33,34] The task of sentiment analysis can be a binary classification of positive and negative opinions,[35,36] as well as extracting the strength of the sentiment.[37,38]

Wehrmann et al[39] proposed a language agnostic sentiment analysis model that can be built on text from different languages. They state that, unlike other cross language sentiment analysis studies that are mostly based on translating the data as the first step, they do not use translation in the process. They threat all of the tweets from 4 different languages as one single corpora and perform binary sentiment classification on manually labeled tweets in these languages. They compare 4 different deep neural network architectures, as well as SVM-based approaches in terms of performance. The input text is processed as a sequence of word embeddings or character based representations, and the Conv-Char-S model performs the best.

Shi et al[40] describe sentiment classification on Weibo (a Chinese microblogging website) data which contains 5000 weibos for each of positive, negative and neutral categories. They compare the effect of selecting different features, as well as various ML algorithms. CNN, SVM, and Naive Bayes models are used in the experiments, and CNN achieves best in terms of precision, recall, and F-measure scores.

Mittal and Goel[41] used Twitter sentiment analysis for the evaluation of machine learning techniques for predicting stock market prices. They use a movie review dataset for training. They compare Linear Regression, Support Vector Machine, Multi Layer Perceptron Neural Network, and LSTMs. LSTMs perform best in their experiments.

Thelwall et al[42] analyzed Twitter data for sentiment analysis correlated with specific events that users react by posting about them. Melamud et al[43] used 6920 sentences as training and 872 sentences as test set for binary (positive/negative) sentiment classification. They aimed to show the performances of different word embedding approaches. They averaged the word embeddings of the words in a sentence to represent the sentence and used an L2-regularized logistic regression classifier for this task.

3.2 | Deep learning

Deep learning is a subfield of machine learning and an adaptation of neural networks. Machine learning algorithms require the distinctive features of the problem extracted and presented. This is seen as the weakest point of machine learning since it requires mostly hand crafted solutions to be developed for each problem domain and a major obstacle in reaching artificial intelligence.[44] On the other hand, a deep learning network “learns” these features on its own from very large datasets without explicitly requiring any feature extracting processes.

Deep learning presents a multi-layered deep architecture for evaluating relationships between input data such as images or text and the results. This deep architecture, instead of performing feature extraction in a separate step, can handle a large number of variables with a single learning algorithm and can extract features.

The reasons for the popularity of this innovative approach in recent years can be summarized as follows:

- In recent years, Graphical Processing Units (GPUs) have enabled powerful and efficient computation and faster processing at lower costs.
- The size of the datasets used for training has increased significantly.
- Recent research on machine learning, data science, and computing has shown significant progress.

3.3 | Multilayer perceptron (MLP)

MLP is the basic type of Artificial Neural Network with at least one hidden layer. MLPs have been widely used in the machine learning applications for classification problems even before the deep learning era.[45,46] MLPs can be considered as a version of a logistic regression classifier.[47]

MLP is a subclass of Feedforward Artificial Neural Networks[48] that transforms the feature values of the input data into linearly separable pre-defined number of spaces where each layer is fully connected. A series of operations is applied on the layers of an MLP starting from the feature values of the input data in one direction toward the output neurons. In a one-hidden-layer MLP, two layers exist other than the input layer. Each of these layers has their own weight matrices, which are usually represented as $W^{(i)}$ where i indicates the layer number in the network.

We can formulate a one-hidden-layer MLP as a function $f : R^D \rightarrow R^L$, where D is the size of input and L is the size of output.[46] Denoting the input feature vector as x, $f(x)$ is as follows:

$$f(x) = G\left(b^{(2)} + W^{(2)}\left(s\left(b^{(1)} + W^{(1)}x\right)\right)\right).$$

Here, $b^{(1)}$ and $b^{(2)}$ are bias vectors and s and G are activation functions.

3.4 | Convolutional neural networks (CNN)

CNN is one of the most popular deep learning model in image classification and image recognition tasks.[49-51] They are also used in solving many other image or text based machine learning problems.[52-54] Usually grid like representation of data such as images in pixels, words in a text document are used as input features.

CNN is another form of MLP that substitutes the hidden layer with a number of layers called as convolutional, pooling, Rectified Linear Units (ReLU), and fully connected layers. The convolutional layer calculates a dot product of its weights and the layer it is connected to. The output of the convolutional layer is reduced by size with the operation of pooling layer, which creates subsamples of the convolutional layer's weights.[55] ReLU is a function $f(x) = \max(0, x)$ that is applied to elements of the previous layer as element-wise. A fully connected layer takes input from the previous layer and calculates the scores for each class.

3.5 | Long short-term memory (LSTM)

LSTM is a gradient based model developed for solving the problem of decaying error backflow in the backpropagation process of recurrent networks.[56] In the basic form of RNN, which is usually called as Vanilla RNN, the input is the current training example with the previous state of the network so that, in a way, it memorizes all of the previous states.

For inputs elements of $\{X^{(0)}, \ldots, X^{(T)}\}$ where $x^{(t)} \in R^{(m)}$ and $h^{(t)} \in R^{(n)}$ as the hidden layer of the RNN for time t,

$$h^{(t)} = f\left(Wh^{(t-1)} + Vx^{(t)}\right).$$

Here, V and W are shared weight matrices and f is a nonlinear activation function such as tanh.[57]

This repeated usage of previous hidden states in the calculation of the hidden states for new time steps causes the well-known vanishing gradient problem. LSTM solves this problem by adding gates such as a forget gate to prevent the network from depending too much on the same first weights. More specifically, an LSTM has input, output, and forget gates that selectively applies read, write, and reset operations on the cells.[58] This architecture is an improvement over Deep Neural Networks, which enables better modeling than RNNs in tasks like speech recognition and handwriting recognition.[58,59]

3.6 | Bi-directional long short time memory (BiLSTM)

BiLSTM is a modification of the LSTM architecture for training in both positive and negative time directions. In other words, BiLSTM is an improved version of LSTM in which the input is simultaneously scanned both in order and in reverse order.[60]

In prediction applications that use sequential data as input, most recent elements of the sequence are important as well as all of the previous elements depending on the nature of the task. For example, in speech recognition related tasks such as in the works of Weninger et al,[61] Wöllmer et al,[62] and Geiger et al,[63] most recent audio frames are very important for the model to recognize the word in that particular time step but the prediction also needs to make use of the frames from the previous time steps. This can be referred as long-term dependency.

BiLSTM is shown to perform better than other models in other prediction studies that use temporal data. Wöllmer et al[64] used BiLSTM for emotion recognition from speech and facial expressions. Singh et al[65] reported that BiLSTM successfully models different actions given in a video dataset (MERL Shopping Dataset), which contains various shopping actions such as “inspect product” or “hand in shelf.”

BiLSTM has two parallel layers propagating in both directions.[66] The final output of such a network would be the concatenation of the output of these two layers. With ${h_F}^{(t)}$ as the hidden state of the forward propagating layer and ${h_B}^{(t)}$ as the hidden state of the backward propagating layer, $h^{(t)}$ the hidden state of the BiLSTM can be calculated as follows:

$$y_t = \left[{h_F}^{(t)} \oplus {h_B}^{(t)}\right].$$

y_t is the output of each time step. This hidden state value is decoded by a linear layer and a softmax layer.[67]

3.7 | Word embeddings

Word embeddings is a preferred technique for transforming text into a vector in text classification, document clustering, speech tagging, entity identification, emotion analysis, and many other natural language processing tasks. These distributed representations aim to transform the words into vectors in order to relate the similarity between the vectors to the semantic similarity between the words. Word2vec[8] and GloVe[7] are two of the best known word embeddings approaches.

In text classification tasks such as sentiment analysis, word embeddings created with word2vec are used to improve performance and obtain more accurate classification results. The word2vec model shown in Figure 1 has two architectures: Continuous Bag of Words (CBOW) shown in Figure 1B and skip-gram shown in Figure 1A. Although the CBOW model is reported as several times faster to train in the literature, it is observed that the skip-gram model works better in predicting non-frequent words. So, we prefer the skip-gram model for obtaining pre-trained word embeddings (PWE) in this study.

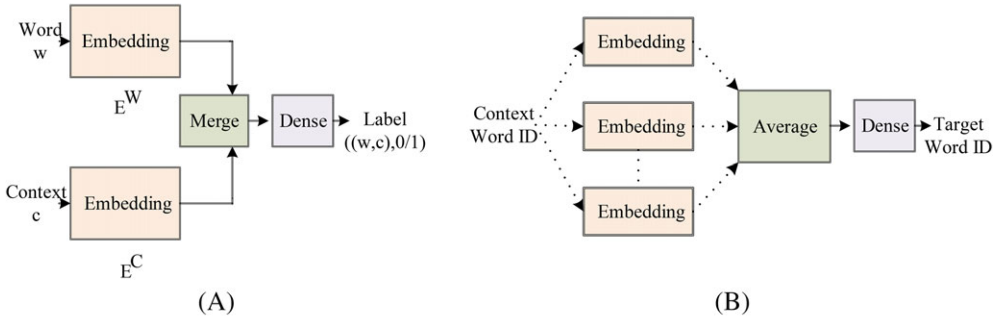

FIGURE 1 Skip-gram and CBOW word2vec models. A, Skip-gram. B, Continuous bag of words

The skip-gram model tries to predict the words around the center of a given word, ie, context words. Figure 1A shows a classifier that takes a word vector and a context vector as inputs. The classifier in Figure 1A is first trained to learn and then to predict the actual label of a review, which is 0 if the sentiment polarity is negative and 1 if the polarity is positive.[68]

In the skip-gram word2vec model, each word $w \in W$ is represented by a word vector, $E^W \in R^D$, and similarly, each context $c \in C$ is represented as a context vector, $E^C \in R^D$, where d is the vector size (embedding size), W is the word, and C is the contextual dictionary. Inner multiplication is performed between words and vector representations of contexts. The output of this trained network is the weight of the word layer. Because of the very low computational complexity, the skip-gram model can calculate very accurate high-dimensional word vectors from very large datasets.[69]

The parameters selected for creating the word embeddings in this paper are guided by the work of Lai et al on creating the word embeddings.[70]

4 | EXPERIMENTS

4.1 | Evaluation environment and dataset

The training and tests conducted on this study were performed on a server running Ubuntu 14.04 operating system with an Intel Xeon X5570 2.93-GHz processor and 98 GB of memory.

A Java based web scraping tool has been developed to download the movie reviews for training word vectors.[71] User and editor reviews were collected from the website www.beyazperde.com. Positive and negative reviews were labeled according to the scores given by the user. Digits, punctuation, and various icons have been removed from the downloaded dataset that contains approximately 13 million words and 80 thousand unique words. Table 1 shows the distribution of the reviews according to user scores. While we use all of the dataset for obtaining the pre-trained word vectors, we use 40,617 reviews for training the deep learning models. We randomly split the training set into the training and validation sets with 32,493 and 8,124 samples, respectively. The reviews that received 0 and 1 scores were considered as negative and a score of 5 was considered positive. The dataset is available at http://buyukveri.firat.edu.tr.

The T-SNE visualization of positive and negative word vectors on two dimensions is shown in Figure 2. Blue points indicate positive words and red points indicate negative words.

4.2 | Deep learning models

TensorFlow (tensorflow.org) and Keras (keras.io) libraries, which are developed with the Python language, have been chosen for building and testing the deep learning models. These open source tools, highlighted by their ability to reduce complexity and ease of use, gave successful results. In this study, we evaluate and compare the following algorithms and models.

4.2.1 | Pre-trained word embeddings (PWE)

Word embeddings are an important input for training Deep Neural Networks. In order to improve performance and obtain more accurate classification results in this study, we have trained deep neural networks using 100-dimensional word vectors to investigate the performance impact of using PWE. Word vector embeddings used in this paper were obtained by applying a skip-gram word2vec model to sentiment classification of movie reviews.

TABLE 1 Polarity distribution on the movie review dataset

	Very Positive	Positive	Neutral	Negative	Very Negative	Total
Number of reviews	21.623	66.946	14.425	33.655	20.167	156.816
Score	5	4	3	2	0, 1	

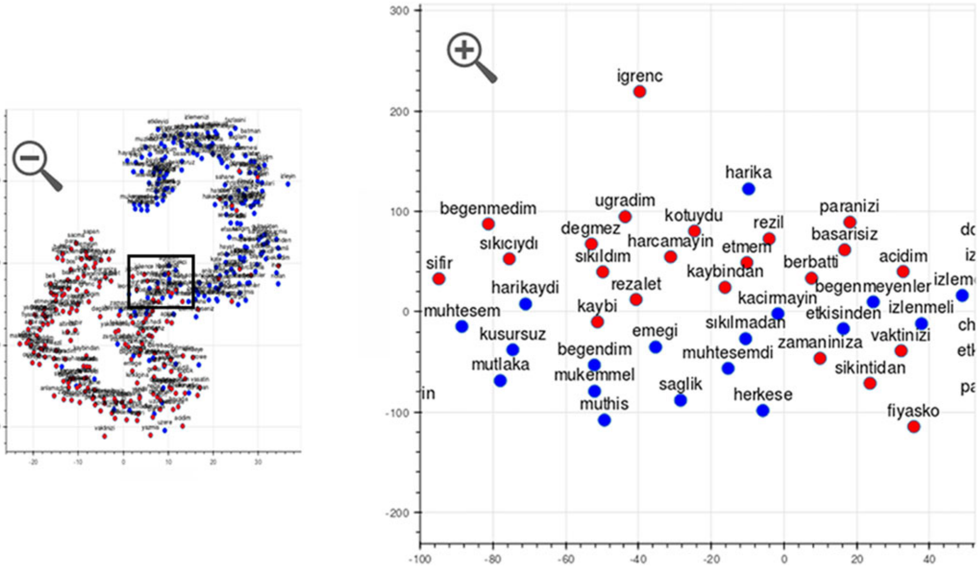

FIGURE 2 The visual overview of positive and negative word vectors in the dataset

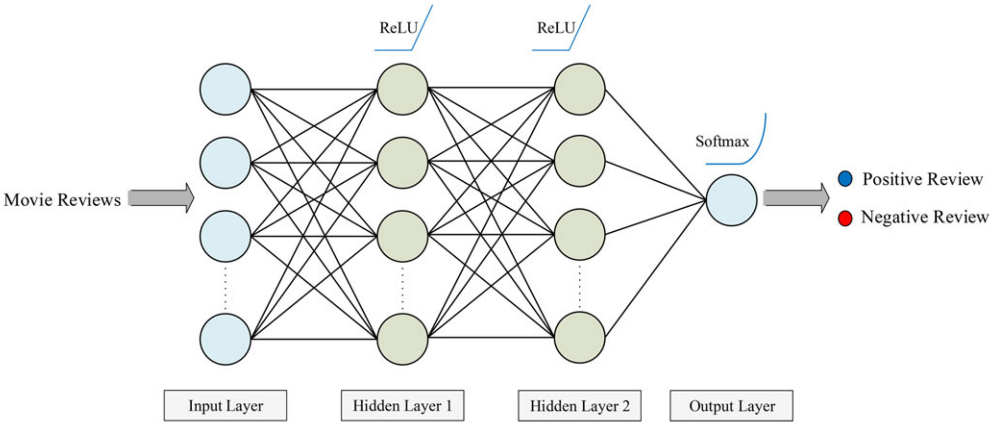

FIGURE 3 The structure of the MLP model

4.2.2 | MLP

In the MLP model, we build and train a simple network on top of TensorFlow for sentiment classification. The model consists of one input layer, two hidden layers, and an output layer. We first convert the reviews into the word vectors and create the vocabulary using bag of words. We use the 30,000 most frequent words as input and ignore the uncommon words in the vocabulary since they have little effect on the predictions. We add two hidden layers to the network with the ReLU activation function. Finally, we use two output units with the Softmax activation function to predict whether the input review belongs to a positive or negative class. The network structure is shown in Figure 3.

4.2.3 | CNN

CNNs are composed of a stack of layers like MLPs but differ from MLPs with the type of hidden layers placed in the model. Three basic types of layers are used to construct a CNN model or LeNet[72]: Convolutional Layer, Pooling Layer, and Fully Connected Layer. Convolutional Layers are locally connected. There are two pooling operations that can be used in the Pooling Layer, average pooling, and max-pooling. The structure of the one-dimensional CNN model used for sentiment classification in this study is inspired by literature[9,24-26] and given in Figure 4.

In this model, each word in the sentence is first replaced by the embedding vector and obtained a sentence matrix expressed as $M \in R^{SXD}$, where S represents the maximum sentence length and D represents the word vector size. The filter size indicates how many words are to be shifted (convolve) in one sentence matrix. The ReLu activation function, $f(x) = max(0, x)$, is applied to all convoluted values. As a result, as many feature maps as the number of filters or kernels are obtained. The most specific information is extracted by applying maximum pooling to each feature map. We add a dropout layer after PWE and before softmax layer to avoid overfitting. Finally, the fully connected layer calculates two points as positive and negative as output.

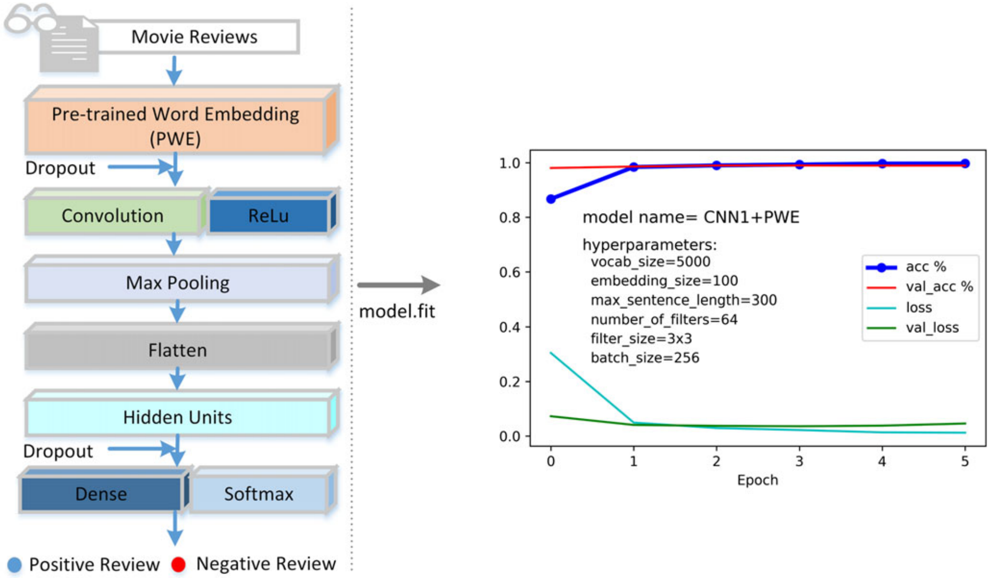

FIGURE 4 CNN model architecture

Figure 2B presents the training results of a model used in this study with the best hyperparameters selected according to the dataset and the model type. The figure shows the accuracy and loss values for each epoch of the training and validation steps of the CNN1 model, which contains only one convolution layer and one pooling layer. The model given in Figure 2 also uses pre-trained word embeddings and therefore called CNN1 + PWE. The model is trained by minimizing the mean square error between desired and actual output values. The other CNN architecture used in this study is called the CNN2 model, which consists of two convolutional layers and two pooling layers.

4.2.4 | LSTM

In deep neural network applications, overfitting and vanishing gradients are two main problems. The dropout method is used to prevent the problem of overfitting caused by the increase in the number of parameters in the hidden layer.[73] Recurrent Neural Networks (RNN), which are widely used in natural language processing, can predict the next word from the previous words given in a text. Like traditional neural networks, RNNs also use back-propagation. During backpropagation, the gradients tend to be zero, which is called the vanishing gradient problem. LSTM architecture is presented for solving this problem.[56] In this study, we investigate the performance of LSTM architecture for sentiment classification.

Figure 5 shows the structure of the LSTM model used in this study on the left and the accuracy and loss results of the model with best hyper-parameters on the right. The LSTM layer tries to find out whether a movie review given is positive or negative by training a many-to-one RNN. The reviews, which are labeled as 1 or 0 depending on whether they are positive or negative, have different sentence lengths. However, for the training step, the maximum sentence length is set as 300 words. Longer reviews are truncated to the maximum sentence length, whereas shorter reviews are “zero-padded,” that is, they are filled with zeroes up to the maximum sentence length. We also add a dropout after the LSTM layer to prevent overfitting. Figure 5 (right) shows how the loss is reduced against the increase in the accuracy for each epoch for the training and validation stages.

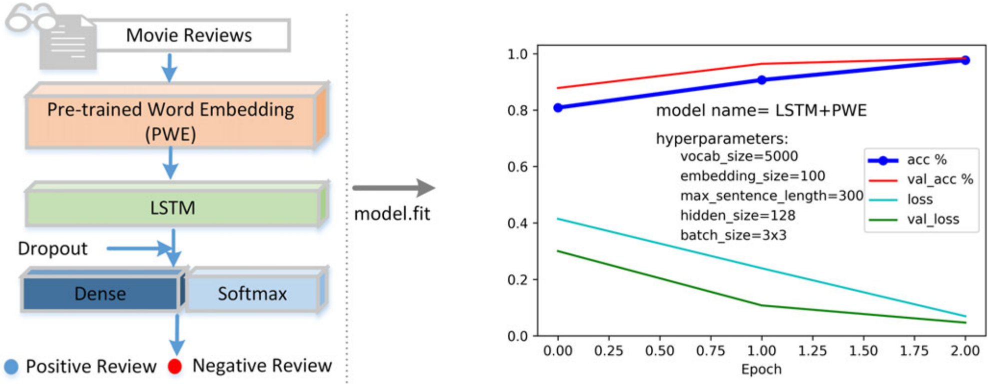

FIGURE 5 LSTM model architecture

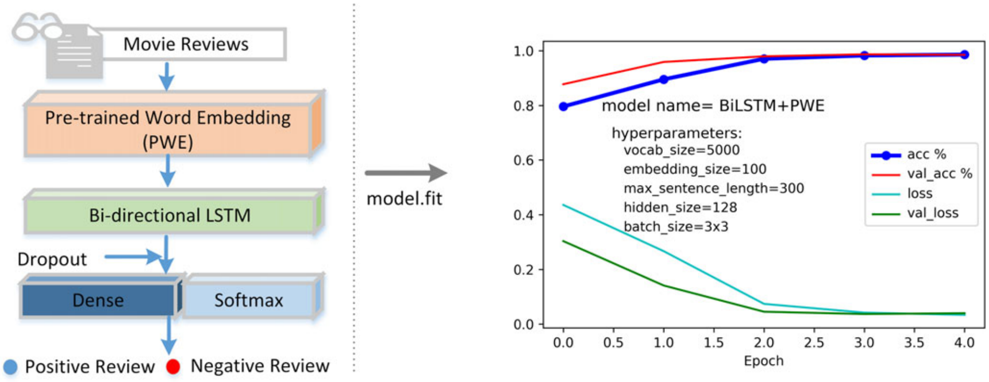

FIGURE 6 BiLSTM model architecture

4.2.5 | BiLSTM

The BiLSTM model has two hidden parallel layers in two directions; forward and backward passes propagate through these layers. For this study, we create a BiLSTM model with two LSTMs stacked on the top of the network, reading the input reviews from the word embeddings. While one LSTM reads the words from left to right, other LSTM reads the words from right to left. The extracted features of each review are passed over forward and backward LSTM networks and fed into the dense layer for classification using the softmax activation function. Dropout is added before the softmax layer to prevent overfitting. The accuracy and loss results for the training and validation steps of the BiLSTM model are given in the chart in Figure 6. The results show that the model performs highly successfully in sentiment classification task.

4.2.6 | CNN-LSTM

Although the CNN model is good at extracting local features, it proves weak in learning sequential data. The CNN-LSTM model architecture used to solve this problem is given in Figure 7 (left), whereas Figure 7 (right) shows the model parameters along with the training results. Unlike the C-LSTM model proposed by Zhou et al,[23] after the convolution layer, the maximum pooling layer is added since the maximum pooling layer reduces the number of parameters and the computational time.

Appropriate parameter selections were made on the model according to the dataset, and the word embeddings were obtained from the same comprehensive movie reviews dataset. The model consists of a combination of single-layered hierarchical CNN and a sequential LSTM architectures. As CNN learns local features from the input reviews, LSTM learns long-term dependencies and processes them sequentially. First, word vectors were trained using the skip-gram word2vec algorithm on the movie reviews dataset. Each word in the input review (here we call a sentence) is represented by these pre-trained word vectors. Thus, the first layer of the model is the word representations or embeddings. In the convolution layer, features are extracted and element-wise multiplication is performed on these features by the corresponding filter. Features that were multiplied in the pooling layer are reduced by the maximum pooling process. The LSTM layer learns the features it receives after the convolution and pooling layers and predicts whether a given review is positive or negative. We use dropout of 0.3 before the convolutional layer and after the LSTM layer to overcome overfitting. The accuracy and loss results for the training and validation steps of the CNN-LSTM model are given in the chart in Figure 7.

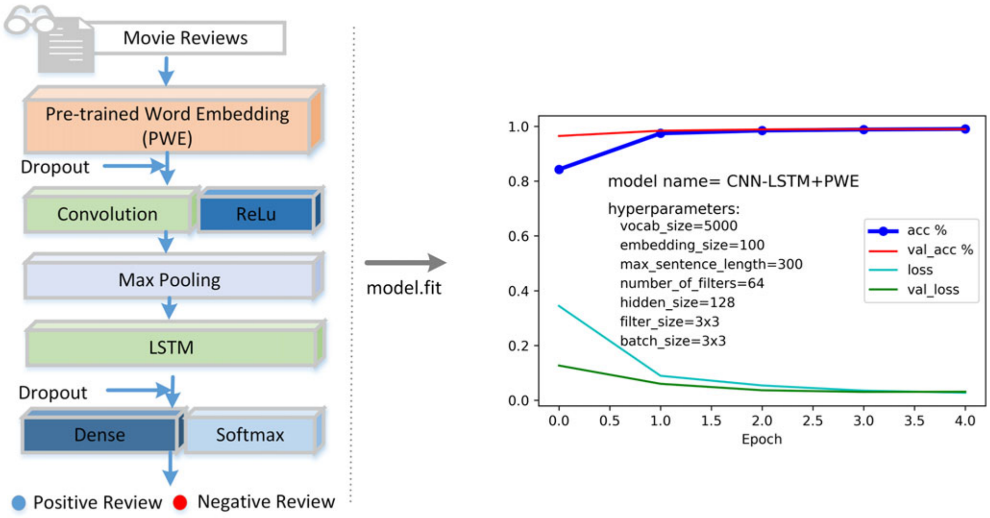

FIGURE 7 CNN-LSTM model architecture

As the deep neural network is being trained, a loss function is used, which serves as an indicator to measure how far the actual output is from the predicted output. The categorical cross entropy function was used to calculate loss values for the training.

5 | RESULTS AND DISCUSSION

In this paper, we trained and evaluated several neural networks for a varying number of epochs and batch sizes. We trained our models using the Adam[57] optimizer with a learning rate of 0.001 and batch sizes of 32, 64, 128, and 256. We optimized the number of training iterations or epochs automatically using an early stopping technique. The early stopping technique supported by Tensorflow determines when to stop training the model by monitoring the validation error. In this way, it helps to avoid strong the overfitting of hyper-parameters by tuning epoch numbers. We also use dropout, a regularization method to overcome overfitting for all models.

During training and validation processes, overfitting was observed for the experimented models. Figure 8 shows the training accuracy and loss results for the Deep Learning models experimented in this study. The results show that, when the models are overfitted, the training-validation accuracies and training-validation losses diverge, which indicates the overfitting.

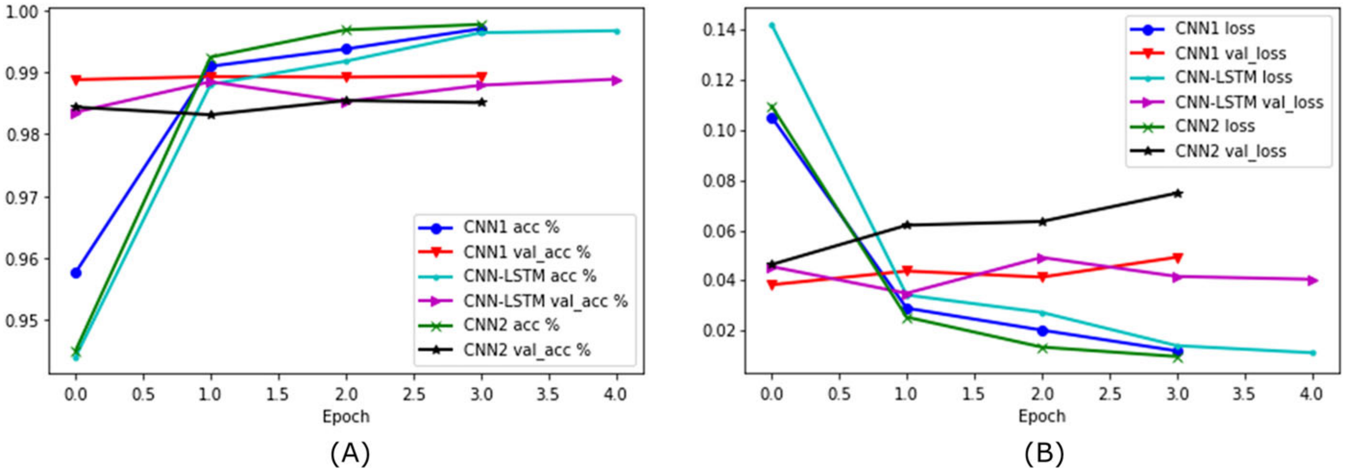

FIGURE 8 Training accuracy A and loss B results for overfitting models

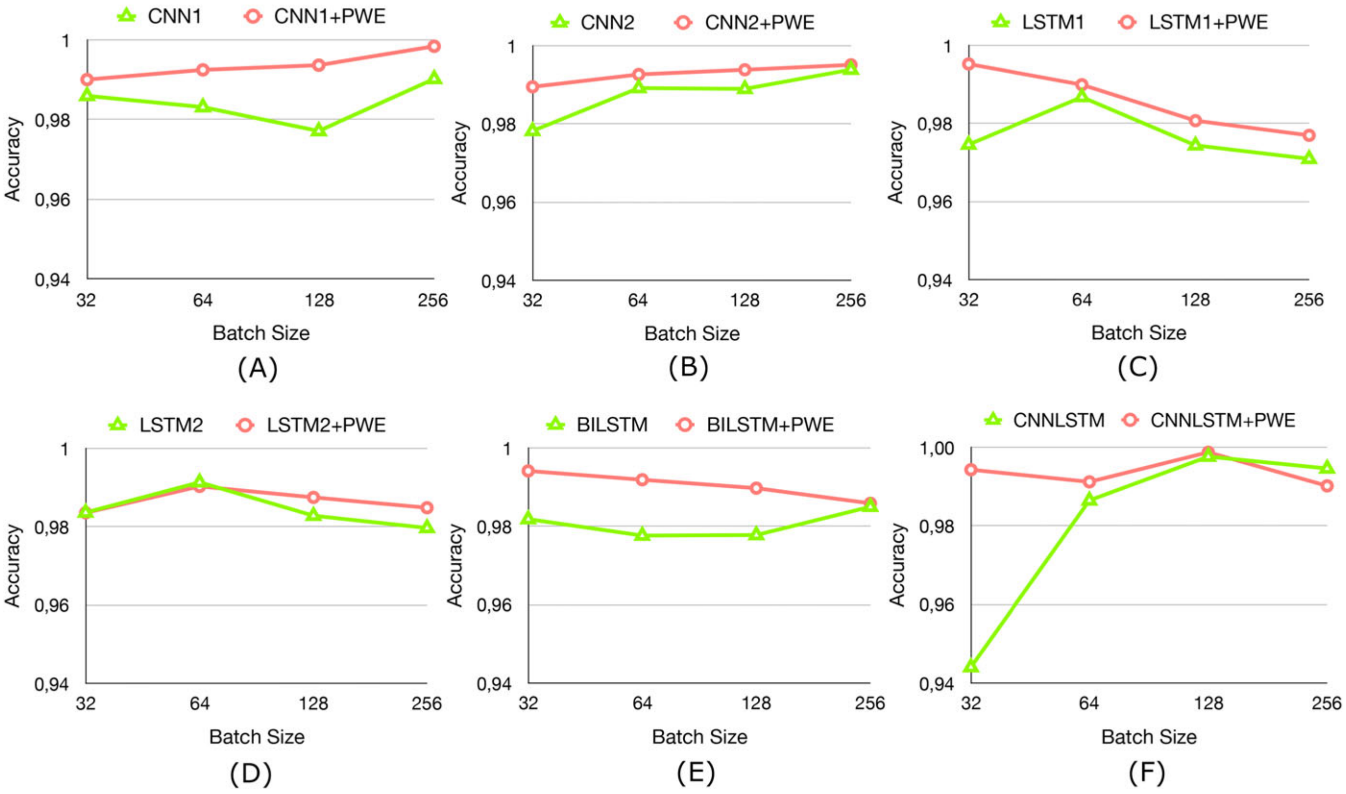

FIGURE 9 Effect of PWE on accuracy for (A) CNN1, (B) CNN2, (C) LSTM1, (D) LSTM2, (E) BILSTM, and (F) CNNLSTM

PWE can capture syntactic and semantic regularities in language.[74] With PWE, deep neural networks achieve good performance in NLP tasks.[75] In this study, we experimented various deep learning models with PWE and analyzed the effects for sentiment classification tasks. It can be observed from the experiments and the results, which are given in Figure 9 that the models initialized with PWE gives better training results than the models without PWE. We also observed that one and two layer CNN models without PWE exhibit excessive overfitting and the use of PWE helps prevent it.

In this study, the effect of larger batch sizes on accuracy and loss values depending on the dataset is investigated. Lower batch size has caused overfitting for all models, especially CNN models. Therefore, we conclude that, if PWE is not used, batch sizes of 128 and 256 are more appropriate.

Table 2 outlines the performance improvements obtained by incorporating PWE over deep models without PWE. As seen in the table, incorporating PWE with the models results in accuracy improvements for all models. However, most significant increases are observed in CNN1 and CNN-LSTM with 1.65% and 0.05%, respectively. The total computational time or cost is very high for BiLSTM and two-layer LSTM2 models. Considering the running time advantage and cost accuracy, CNN1 and CNN-LSTM models with PWE have demonstrated more accurate binary sentiment classification results.

Figure 10 shows the accuracy results according to various batch sizes for all models studied in this paper. We can draw the following conclusions:

- One layer CNN1 + PWE performed better than two layers CNN2 and one layer CNN1.
- One layer LSTM1 + PWE performed better than two layers LSTM2, one layer LSTM1, and BiLSTM.
- For one layer networks with PWE, CNN1 performed better than LSTM1, whereas for two layer networks with PWE, LSTM2 performed better than CNN2.
- CNN-LSTM+PWE performed better than all models.
- MLP performed more poorly than all deep learning models.
- CNN1 achieved better results than CNN2.

Table 3 gives sample reviews from the dataset and outputs for the CNN-LSTM + PWE model. The predicted value of the model is rescaled into the range between 0 and 1. If the value is close to 1, the predicted label is 1; otherwise, it is 0. It can be observed that the predicted labels match the actual labels of the movie reviews.

TABLE 2 Performance improvements for deep models with PWE

Models	Figure 8	Max Performance Improvement with PWE, %	Batch Size	Training and Test time, s
CNN1	a	**1.654**	128	310
BILSTM	b	0.014	64	7514
LSTM1	c	0.020	32	955
CNN2	d	0.011	32	520
CNNLSTM	e	**0.050**	32	481
LSTM2	f	0.005	256	6210

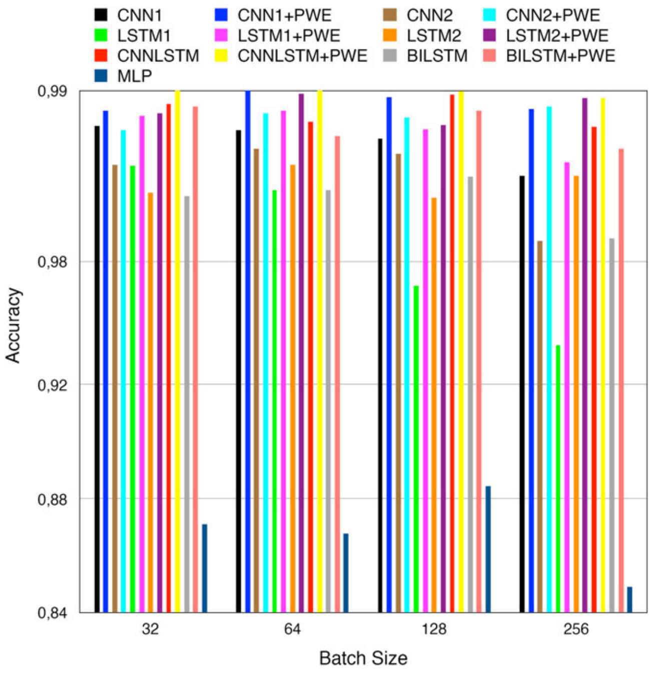

FIGURE 10 Classification accuracy results for different batch sizes

TABLE 3 Model output for some random test sentences

Predicted Value	Predicted Label	Actual Label	Sentence/Review
0.787164	1	1	film gercekten tam bi.. superdi ya..
0.000068	0	0	berbat tamamen zaman kaybi
0.820993	1	1	muthis bir film bayildim hic kovalamaca dovus eglence kahkaha . bu filmin serisi bitmesin. gercekten harika .
0.818120	1	1	muhtesem senaryo muhtesem oyunculuk muhtesem yonetmen . . kill bill i izlemeyen cok sey kaybetmis …
0.000029	0	0	asiri dandik bi film arkadaslar vakit kaybi olur sizin icin hele aile sakin ha sakin izlemeyin . tuhaf bi film ya .
0.426062	0	1	kotu amerikan. yaptigi icin oscar alacak demistim .. izleyince anladim tam tersine malesef oscari kaybetmis . demekki oscar guzel muhtesem .
0.000132	0	0	Hayal kirikligi . izleyin gorun .

TABLE 4 Test accuracy results for 4000 samples

Model (PWE)	Number of False Classified Samples	Test Accuracy, %
MLP	869	78,27
CNN1	96	97,60
CNN2	95	97,62
LSTM1	137	96,57
LSTM2	139	96,52
BILSTM	129	96,77
CNNLSTM	**77**	**98,07**

We also use a separate dataset consisting of 4000 movie reviews for testing the trained models. The test set contains 2000 reviews each for positive and negative polarities. The test accuracy results for each model studied in this paper are given in Table 4.

Scaling Deep Learning can be achieved by using various big data technologies. The paper deals with sentiment classification of movie reviews. However, the Deep Learning models developed in this study can be used to classify a very large number of reviews. Once the DL model is generated, it takes only a fraction of time to process new data using this model. Distributed file systems such as HDFS or NoSQL databases, eg, HBase or MongoDB, can be used to store very large amounts of data. A scalable data storage system based on such technologies can be utilized to store big data, and the system can be extended to read big data from these scalable storages. Once the data are read, the system would employ the already trained DL model to classify them. Therefore, scaling Deep Learning models require (a) training the DL model to be used on a training set and (b) providing access to the big data stored on distributed data storage systems such as distributed file systems or NoSQL databases for processing using deep models.

6 | CONCLUSIONS AND FUTURE WORK

In this study, sentiment classification was performed on Turkish movie reviews collected from the site “www.beyazperde.com.” The whole dataset consists of 44,617 samples including positive and negative reviews. We use 4000 samples from the dataset for testing the models. We trained our models using the Adam[76] optimizer with batch sizes of 32, 64, 128, and 256. We optimized the number of epochs automatically using the early stopping technique. The success of sentiment classification was compared on CNN, LSTM, and the combination of these models CNN-LSTM, BiLSTM, and MLP models. Additionally, 100-dimensional pre-trained word embeddings (PWE) were obtained by applying a skip-gram word2vec model to a larger movie review corpus. The effects of the PWE on deep learning models are discussed. The results show that, with PWE, all models performed better than the models without PWE. Furthermore, the use of PWE reduced overfitting considerably. Lastly, the hybrid model of CNN-LSTM with PWE achieved the best results. On the other hand, MLP ranked the last among all models. In terms of overall running time, CNN1 and CNN-LSTM models demonstrated the best performance.

This paper presented a binary classification study for the Turkish language on movie reviews. We are planning to extend the sentiment classification in other domains by crawling data from various web sites in Turkish such as product reviews, hotel reviews, and book reviews. We will investigate the effect of using cross domain word embeddings on sentiment classification. Also, we plan to study multiclass classification using different Deep Learning models on news data, which includes classes such as economy, sports, health, and technology.

ORCID

Betül Ay Karakuş http://orcid.org/0000-0002-3060-0432

REFERENCES

1. Cui H, Mittal V, Datar M. Comparative experiments on sentiment classification for online product reviews. Paper presented at: 21st National Conference on Artificial Intelligence; 2006; Boston, MA.
2. Qu Y, Shanahan J, Wiebe J. Exploring attitude and affect in text: Theories and applications. Paper presented at: AAAI Spring Symposium; 2004; Palo Alto, CA.
3. Socher R, Perelygin A, Wu J, et al. Recursive deep models for semantic compositionality over a sentiment treebank. Paper presented at: Conference on Empirical Methods in Natural Language Processing; 2013; Seattle, WA.
4. Pang B, Lee L. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. Paper presented at: 43rd Annual Meeting on Association for Computational Linguistics; 2005; Ann Arbor, MI.
5. Deng L, Yu D. Deep learning: methods and applications. *Found Trends® Signal Process*. 2014;7(3-4):197-387.
6. Sarkar D, Bali R, Sharma T. Analyzing movie reviews sentiment. In: *Practical Machine Learning with Python*. New York, NY: Springer; 2018:331-372.
7. Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation. Paper presented at: Conference on Empirical Methods on Natural Language Processing; 2014; Doha, Qatar.
8. Mikolov T, Sutskever I, Chen K, Corrado GS, Dean J. Distributed representations of words and phrases and their compositionality. Paper presented at: Annual Conference on Neural Information Processing Systems; 2013; Stateline, NV.
9. Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P. Natural language processing (almost) from scratch. *J Mach Learn Res*. 2011;12:2493-2537.
10. Liu B. Sentiment analysis and subjectivity. In: Indurkhya N, Damerau F, eds. *Handbook of Natural Language Processing*. Boca Raton, FL: CRC Press; 2010:627-666.
11. Pang B, Lee L. Opinion mining and sentiment analysis. *Found Trends® Inf Retr*. 2008;2(1-2):1-135.
12. Catal C, Nangir M. A sentiment classification model based on multiple classifiers. *Appl Soft Comput*. 2017;50:135-141.
13. Agarwal B, Mittal N. Machine learning approach for sentiment analysis. In: *Prominent Feature Extraction for Sentiment Analysis*. Cham, Switzerland: Springer; 2016:21-45.
14. Kalaivani P, Shunmuganathan KL. Sentiment classification of movie reviews by supervised machine learning approaches. *Indian J Comput Sci Eng*. 2013;4(4):285-292.
15. Wawre SV, Deshmukh SN. Sentiment classification using machine learning techniques. *Int J Sci Res*. 2016;5(4):819-821.
16. Bhadane C, Dalal H, Doshi H. Sentiment analysis: measuring opinions. *Procedia Comput Sci*. 2015;45:808-814.
17. Akkarapatty N, Muralidharan A, Raj NS, Vinod P. Dimensionality reduction techniques for text mining. In: Bhatnagar V, ed. *Dimensionality reduction techniques for text mining*. Hershey, PA: Information Science Reference; 2016:49.
18. Hinton GE, McClelland JL, Rumelhart DE. Distributed representations. In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge, MA: MIT Press; 1984:77-109.
19. Roukos S, Quin J, Ward T. Multi-lingual text leveling. Paper presented at: 17th International Conference on Text, Speech and Dialogue; 2014; Brno, Czech Republic.
20. Mikolov T, Deoras A, Kombrink S, Burget L, Černock'y J. Empirical evaluation and combination of advanced language modeling techniques. Paper presented at: 12th Annual Conference of the International Speech Communication Association; 2011; Florence, Italy.
21. Ouyang X, Zhou P, Li CH, Liu L. Sentiment analysis using convolutional neural network. Paper presented at: IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing; 2015; Liverpool, UK.
22. Tang D, Qin B, Liu T. Document modeling with gated recurrent neural network for sentiment classification. Paper presented at: Conference on Empirical Methods in Natural Language Processing; 2015; Lisbon, Portugal.
23. Zhou C, Sun C, Liu Z, Lau F. A C-LSTM neural network for text classification. 2015. arXiv Prepr. arXiv1511.08630.
24. Kim Y. Convolutional neural networks for sentence classification. 2014. arXiv Prepr. arXiv1408.5882.
25. Pang B, Lee L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. Paper presented at: 42nd Annual Meeting on Association for Computational Linguistics; 2004; Barcelona, Spain.
26. Hu M, Liu B. Mining and summarizing customer reviews. Paper presented at: 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004; Seattle, WA.
27. Wen Y, Zhang W, Luo R, Wang J. Learning text representation using recurrent convolutional neural network with highway layers. 2016. arXiv preprint arXiv:1606.06905.
28. Hong J, Fang M. Sentiment analysis with deeply learned distributed representations of variable length texts. Stanford, CA: Stanford University; 2015. Technical report.
29. Chen P, Plale B, Aktas MS. Temporal representation for mining scientific data provenance. *Future Gener Comput Syst*. 2014;36:363-378.
30. Jensen S, Plale B, Aktas MS, Luo Y, Chen P, Conover H. Provenance capture and use in a satellite data processing pipeline. *IEEE Trans Geosci Remote Sens*. 2013;51:5090-5097.
31. Aktas MS, Plale B, Leake D, Mukhi N. Unmanaged Workflows: their provenance and use. In: Liu Q, ed. *Data Provenance and Data Management in eScience*. Berlin, Germany: Springer-Verlag Berlin Heidelberg; 2013:59-81.
32. Aktas MS, Astekin M. Provenance aware run-time verification of things for self-healing internet of things applications. *Concurr Comput Pract Exp*. https://doi.org/10.1002/cpe.4263

33. Xu L, Liu J, Wang L, Yin C. Aspect based sentiment analysis for online reviews. In: Park JJ, Loia V, Yi G, Sung Y, eds. *Advances in Computer Science and Ubiquitous Computing*. Singapore: Springer; 2017:475-480.

34. Hemmatian F, Sohrabi MK. A survey on classification techniques for opinion mining and sentiment analysis. *Artif Intell Rev*. 2017;1-51.

35. Thelwall M, Buckley K, Paltoglou G, Cai D, Kappas A. Sentiment strength detection in short informal text. *J Assoc Inf Sci Technol*. 2010;61(12):2544-2558.

36. Thelwall M, Buckley K, Paltoglou G. Sentiment strength detection for the social web. *J Assoc Inf Sci Technol*. 2012;63(1):163-173.

37. Kouloumpis E, Wilson T, Moore JD. Twitter sentiment analysis: The good the bad and the OMG!. Paper presented at: International AAAI Conference on Weblogs and Social Media; 2011; Barcelona, Spain.

38. Pak A, Paroubek P. Twitter as a corpus for sentiment analysis and opinion mining. Paper presented at: International Conference on Language Resources and Evaluation; 2010; Valletta, Malta.

39. Wehrmann J, Becker W, Cagnini HEL, Barros RC. A character-based convolutional neural network for language-agnostic twitter sentiment analysis. Paper presented at: International Joint Conference on Neural Networks; 2017; Anchorage, Alaska.

40. Shi S, Zhao M, Guan JUN, Huang H. Multi-features group emotion analysis based on CNN for Weibo events. Paper presented at: 2nd International Conference on Computer Engineering, Information Science and Internet Technology; 2017; Dubai, UAE.

41. Mittal A, Goel A. Stock prediction using Twitter sentiment analysis. Stanford, CA: Standford University; 2012. http://cs229.stanford.edu/proj2011/GoelMittal-StockMarketPredictionUsingTwitterSentimentAnalysis.pdf

42. Thelwall M, Buckley K, Paltoglou G. Sentiment in Twitter events. *J Assoc Inf Sci Technol*. 2011;62(2):406-418.

43. Melamud O, McClosky D, Patwardhan S, Bansal M. The role of context types and dimensionality in learning word embeddings. 2016. arXiv Prepr. arXiv1601.00893.

44. Sugomori Y. *Java Deep Learning Essentials*. Birmingham, UK: Packt Publishing Ltd; 2016.

45. Bourlard H, Wellekens CJ. Links between Markov models and multilayer perceptrons. Paper presented at: Advances in Neural Information Processing Systems; 1989; Denver, CO.

46. Block HD. A review of perceptrons: an introduction to computational geometry≓. *Inf Control*. 1970;17(5):501-522.

47. LISA Lab. *Deep Learning Tutorial*. Montreal, Canada: University of Montreal; 2014.

48. Gardner MW, Dorling SR. Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. *Atmos Environ*. 1998;32(14-15):2627-2636.

49. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Paper presented at: 26th Annual Conference on Neural Information Processing Systems; 2012; Lake Tahoe, NV.

50. Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: a convolutional neural-network approach. *IEEE Trans Neural Netw*. 1997;8(1):98-113.

51. Zhou L, Li Q, Huo G, Zhou Y. Image classification using biomimetic pattern recognition with convolutional neural networks features. *Comput Intell Neurosci*. 2017;2017(2).

52. Yu Y, Liu F. A two-stream deep fusion framework for high-resolution aerial scene classification. *Comput Intell Neurosci*. 2018;2018.

53. Ciresan DC, Meier U, Gambardella LM, Schmidhuber J. Convolutional neural network committees for handwritten character classification. Paper presented at: International Conference on Document Analysis and Recognition; 2011; Beijing, China.

54. Hu B, Lu Z, Li H, Chen Q. Convolutional neural network architectures for matching natural language sentences. Paper presented at: Advances in Neural Information Processing Systems; 2014; Montreal, Canada.

55. Deng L. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Trans Signal Inf Process*. 2014;3.

56. Hochreiter S, Schmidhuber J. Long short-term memory. *Neural Comput*. 1997;9(8):1735-1780.

57. Ming Y, Cao S, Zhang R, Qu H. Understanding hidden memories of recurrent neural networks. 2017. arXiv Prepr. arXiv1710.10777.

58. Liwicki M, Graves A, Fernàndez S, Bunke H, Schmidhuber J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. Paper presented at: 9th International Conference on Document Analysis and Recognition; 2007; Curitiba, Brazil.

59. Sainath TN, Vinyals O, Senior A, Sak H. Convolutional, long short-term memory, fully connected deep neural networks. Paper presented at: IEEE International Conference on Acoustics, Speech and Signal Processing; 2015; Brisbane, Australia.

60. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. *IEEE Trans Signal Process*. 1997;45(11):2673-2681.

61. Weninger F, Geiger J, Wöllmer M, Schuller B, Rigoll G. The Munich 2011 CHiME challenge contribution: NMF-BLSTM speech enhancement and recognition for reverberated multisource environments. Paper presented at: Machine Listening in Multisource Environments; 2011; Florence, Italy.

62. Wöllmer M, Eyben F, Schuller B, Rigoll G. A multi-stream ASR framework for BLSTM modeling of conversational speech. Paper presented at: IEEE International Conference on Acoustics, Speech and Signal Processing; 2011; Prague, Czech Republic.

63. Geiger JT, Zhang Z, Weninger F, Schuller B, Rigoll G. Robust speech recognition using long short-term memory recurrent neural networks for hybrid acoustic modelling. Paper presented at: 15th Annual Conference of the International Speech Communication Association; 2014; Singapore.

64. Wöllmer M, Metallinou A, Eyben F, Schuller B, Narayanan S. Context-sensitive multimodal emotion recognition from speech and facial expression using bidirectional LSTM modeling. Paper presented at: INTERSPEECH 2010; 2010; Makuhari, Japan.

65. Singh B, Marks TK, Jones M, Tuzel O, Shao M. A multi-stream bi-directional recurrent neural network for fine-grained action detection. IEEE Conference on Computer Vision and Pattern Recognition; 2016; Las Vegas, NV.

66. Yao Y, Huang Z. Bi-directional LSTM recurrent neural network for Chinese word segmentation. Paper presented at: International Conference on Neural Information Processing; 2016; Kyoto, Japan.

67. Chiu JPC, Nichols E. Named entity recognition with bidirectional LSTM-CNNs. 2015. arXiv Prepr. arXiv1511.08308.

68. Gulli A, Pal S. *Deep Learning with Keras*. Birmingham, UK: Packt Publishing Ltd; 2017.

69. Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. 2013. arXiv Prepr. arXiv1301.3781.

70. Lai S, Liu K, He S, Zhao J. How to generate a good word embedding. *IEEE Intell Syst*. 2016;31(6):5-14.

71. Available: https://github.com/galipaydin/beyazperde/

72. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. *Proc IEEE*. 1998;86(11):2278-2324.

73. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: a simple way to prevent neural networks from overfitting. *J Mach Learn Res*. 2014;15(1):1929-1958.

74. Mikolov T, Yih W, Zweig G. Linguistic regularities in continuous space word representations. Paper presented at: Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies; 2013; Atlanta, GA.

75. Lai S, Xu L, Liu K, Zhao J. Recurrent convolutional neural networks for text classification. Paper presented at: 29th AAAI Conference on Artificial Intelligence; 2015; Austin, TX.

76. Kingma DP, Ba J. Adam: a method for stochastic optimization. 2014. arXiv Prepr. arXiv1412.6980.

How to cite this article: Ay Karakuş B, Talo M, Hallaç İR, Aydin G. Evaluating deep learning models for sentiment classification. *Concurrency Computat Pract Exper*. 2018;30:e4783. https://doi.org/10.1002/cpe.4783