IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received March 6, 2020, accepted March 17, 2020, date of publication March 23, 2020, date of current version April 6, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2982538

A Novel Hybrid Deep Learning Model for Sentiment Classification

MEHMET UMUT SALUR[1] **AND ILHAN AYDIN**[2]
[1]Department of Computer Engineering, Harran University, 63290 Şanlıurfa, Turkey
[2]Department of Computer Engineering, Fırat University, 23190 Elazığ, Turkey

Corresponding author: Mehmet Umut Salur (umutsalur@harran.edu.tr)

ABSTRACT A massive use of social media platforms such as Twitter and Facebook by omnifarious organizations has increased the critical individual feedback on the situation, events, products, and services. However, sentiment classification plays an important role in the user's feedback evaluation. At present, deep learning such as long short-term memory (LSTM), gated recurrent unit (GRU), bidirectionally long short-term memory (BiLSTM) or convolutional neural network (CNN) are prevalently preferred in sentiment classification. Moreover, word embedding such as Word2Vec and FastText is closely examined in text for mapping closely related to the vectors of real numbers. However, both deep learning and word embedding methods have strengths and weaknesses. Combining the strengths of the deep learning models with that of word embedding is the key to high-performance sentiment classification in the field of natural language processing (NLP). In the present study, we propose a novel hybrid deep learning model that strategically combines different word embedding (Word2Vec, FastText, character-level embedding) with different deep learning methods (LSTM, GRU, BiLSTM, CNN). The proposed model extracts features of different deep learning methods of word embedding, combines these features and classifies texts in terms of sentiment. To verify the performance of the proposed model, several deep learning models called basic models were created to perform series of experiments. By comparing, the performance of the proposed model with that of past studies, the proposed model offers better sentiment classification performance.

INDEX TERMS Sentiment classification, Turkish tweets analysis, hybrid model, word embedding, deep learning, LSTM, CNN.

I. INTRODUCTION

Human by nature communicates with one another. In the entire human history, communication has been an important element to solve problems and enhance social engagement. However, present-day communication has changed drastically when compared with the olden days' communication. At present, social media has become an important communication tool, used by almost all segments of society [1]. Facebook, Twitter, Instagram, and YouTube are the leading social media applications. However, Twitter comprises of personal blog features that allow instant sharing among users. Individual users, institutions or organizations use Twitter to communicate and to make important decisions. In this respect, Twitter facilitates interactions between users and institutions or organizations.

The wide use of social media offers opportunities for people to take a feedback on situations, events, products and services [2]. These feedbacks are often based on users' experience, which may be positive or negative opinions on products or services. Identifying negative user opinions is critical to the growth of the organizations [3]. These opinions will help organizations to improve their products and services, thereby assisting them to earn more profit. Therefore, it is important to evaluate user feedback collected from social media and websites. Sentiment analysis is effective in revealing the users' opinions (positive, negative, or neutral) about a product or service through text data [1], [4]. The biggest advantage of sentiment analysis is to evaluate the comments shared by users on product or service providers. In sentiment analysis, the sub-processes in Figure 1 are carried out by analyzing the user contents shared with the help of social media.

From past studies on sentiment analysis, deep learning is presently preferred to machine learning algorithms such as Naive Bayes, Support Vector Machines (SVM), Decision

The associate editor coordinating the review of this manuscript and approving it for publication was Aysegul Ucar.

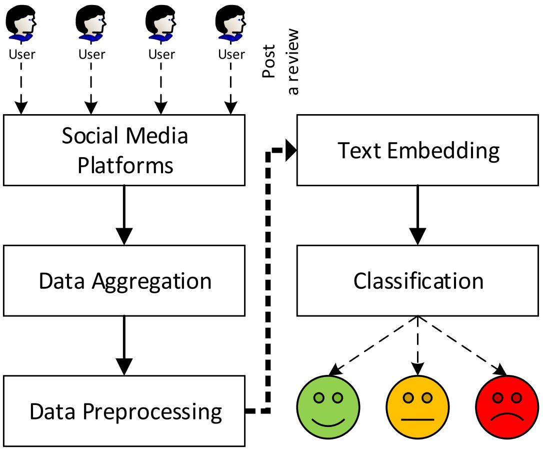


FIGURE 1. **Basic steps of sentiment analysis on social media.**

Trees, Random Forests, frequently used for classification [4]. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are the pioneer of deep learning for sentiment classification. In the present day, deep learning is better than machine learning in sentiment analysis problems owing to huge datasets and low cost of mass production of powerful Graphics Processing Unit (GPU) cards [5]. In many text classification problems, compatibility between the methods to represent texts and their algorithms is significant. With the help of a good text representation and classification algorithm, it is possible to offer high performing classification.

Each deep learning method is characterized by a specific feature, used actively to solve a specific problem or to analyze datasets. On the other hand, the text representation that converts texts to numerical format has also its strengths and weaknesses. The combination of the optimal text representation and the optimal deep learning for the sentiment analysis (i.e., the sentiment classification problem) is important for solving problems and achieving high accuracy. Extracting features from text-based data using different methods contributes to classification performance. Every text representation method (i.e., word embedding, character-level embedding) may be incomplete in the numerical representation of user's sentiment. In line with this problem, we aim to combine the power of different text representations methods and different deep learning models. We propose a new hybrid deep learning model that uses different representations of data and different deep learning model to analyze a dataset from Turkish Twitter messages. Two different strategies were followed in the text data representation, and two different deep learning methods were used with the classification phase. The outcome of the experiments shows that the proposed method offers better classification performance compared with previous studies' performances on the dataset. In addition, many classification experiments have been performed to indicate the effectiveness of the proposed method. This study provides the following contributions:

- A roadmap is presented to embedding Turkish text datasets.
- Deep learning methods with different characters were used for text classification. The classification performance is increased based on this combined usage.
- A new hybrid model is proposed that combines different text representations and deep learning methods. High classification success has been achieved since the proposed model extracts better features and different approaches from the dataset.
- Many experiments are performed to confirm the performance of the proposed model, and this is compared with the methods in the literature.

The rest of the paper is organized as follows: In Section II, previous studies on hybrid studies are explained in detail. Section III discusses the fundamentals of deep learning and metrics of classification performance. Section IV presents the methodology of the proposed method. Based on the experimental environment, Section V presents models and classification results to verify the classification performance of the proposed model. Section VI concludes the paper and discusses future work.

II. LITERATURE

Sentiment analysis has been among the top research priorities for many years since this facilitates important user evaluations in many applications. Within the scope of these studies, it was seen that better classification performances were obtained with hybrid algorithms. Within the scope of these studies, it was discovered that better classification performances were obtained using the hybrid algorithm. Liu *et al.* [6] combined machine learning with deep learning to provide better sentiment classification performance. In their study, the effectiveness of the proposed method is shown on Turkish and Chinese language datasets. Moreover, CNN and LSTM based sentiment analysis were carried out on IMDb comments [7]. In their proposed method, a new approach for emotion analysis is presented using a large number of CNN-LSTM layers that are combined with kernels. After applying CNN to the texts, the sequential features windows feed the LSTM network directly [8]. This method allows LSTM to learn long-term dependencies from higher-order features. Those authors, therefore, combine the strengths of CNN for extracting local features with LSTM for discovering the long-dependency of sentiments. In [9], a new deep learning architecture has been proposed with hybrid CNNs and BiLSTM (H2CBi) features, which combine both CNNs and BiLSTM power. They used two different pre-trained word vectors to obtain different feature vectors given as input to LSTM. Besides the word embedding features, a study [10] has extracted the user and content-based manual features of the dataset collected from the Chinese social media, which is known as Sina Weibo. The obtained features of their study serve as an input to the LSTM network for classification purposes. From their experiments, it is understood that their proposed hybrid method performs better than simple LSTM or conventional machine learning. In contrast, traditional machine learning approaches achieve better classification scores than the deep learning models in Lithuanian languages [11]. The main reason for the success of machine learning methods is because of the complexity of the Lithuanian language since it consists of rich vocabulary, a high number of dialects and difficulty in morphological analysis. A hybrid approach was proposed on machine learning [12]. Their hybrid approach consists of combining both the machine learning using SVM and the semantic orientation approach.

Another study used BiLSTMs to capture long-term dependency information from words and position in the sentence [13]. Their proposed hybrid method combined BiLSTM with CNN. By applying multichannel CNN to LSTM outputs, n-gram features are derived from sentence classification. Unlike our model, their proposed hybrid model uses BiLSTM before multichannel convolution layers. Hashida *et al.* [14] proposed a model that used multi-channel distributed representation, which was a hybrid representation of the word representation for text data. Two channels are used in the text representation, one contains word representations and the second contains the word (noun, verb, adverb, etc.,). Experiments were performed using real travel tweets to evaluate the proposed model, which provided better results than the model from another study [15]. A similar model like ours proposed by using two different data representations together [14]. Moreover, in our model, we simultaneously used two different deep learning methods together. Using CNN for feature extraction from word embedding, Zhou and Long [16] used BiLSTM for the classification stage in Chinese product reviews. From the experiments, the classification success that combined CNN with BiLSTM was better than the basic CNN and BiLSTM classification performance. Instead of using LSTM and CNN consecutively like in their study, we preferred to use LSTM and CNN in a parallel manner on different word representations in our study. In another study [17], a multilevel network of CNN and LSTM was used for sentiment analysis on the dataset using the Tibetan social media application. The features were extracted with the help of the three-layer CNN network. The obtained features are given as an input to the two-layer LSTM network. It was observed that the hybrid deep learning model performed better than CNN and LSTM. A model based on hybrid bidirectional recurrent CNN attention has been proposed [18]. With the help of Word2Vec and the attention mechanism, this model effectively combined BiLSTM with CNN for text classification. In their hybrid model, they used LSTM, CNN, and attention layers sequentially. In another hybrid model, Word2Vec embedding was used to represent the dataset numerically [4]. In the hybrid model created in the study, while CNN is used for feature extraction, LSTM is used for the classification of text-based features. It is discovered that the performance of the proposed model is better than conventional machine learning techniques and simple deep learning models.

Furthermore, Kaladevi and Thyagarajah [19] used one CNN-layer and two-layer stacked LSTM to process sequentially Indian tweets. In their hybrid studies, the features obtained from the CNN layer are given as an input to the LSTM network. Their study used CNN to extract features like other hybrid studies. However, our approach differs from their study since both CNN and BiLSTM for feature extraction were used. Xu *et al.* [20] proposed a hybrid model using a feed-forward artificial neural network and BiLSTM. In their study, BiLSTM is used for feature extraction and a feed-forward neural network is used for the classification phase. In [21], a stack of CNN and LSTM deep learning methods were used and the best performance was obtained than other simple methods. In addition, it is seen that Word2Vec achieves better dataset representation than Word2Seq. Semantic representations, sentiment-based representations, and dictionary-based representations used to encode text [22]. Three attention mechanisms are integrated with the CNN model to extract sentence features. It was reported that the proposed CNN models achieved the best results. Xiao and Cho [23] proposed a hybrid model that consisted of several CNN layers and one RNN layer to process the sequence of character. In their study, CNN was found to be effective in character-level text representations.

When the mentioned studies are evaluated, hybrid deep learning is carried out on a single data representation such as Word2Vec, Glove, and character-level embedding. Moreover, different deep learning methods were used on single data representation. Unlike previous studies, instead of representing texts in a single method, in this study, we propose a novel hybrid deep learning model that feeds on features derived from texts, represented by different embedding methods. The absence of a hybrid study using CNN and LSTM in parallel on different representations of the data offered a motivation to carry out this study.

III. BACKGROUND

In this section, the theoretical background of the methods used in sentiment analysis and text classification problems is briefly mentioned.

A. DEEP LEARNING METHODS

Deep learning is defined as the representation of data in multiple and successive layers. The number of layers in deep learning is an important criterion for representing the depth of the network. Three main developments have an impact on the popularity and effectiveness of deep learning, these include large datasets in training, advancement in hardware resources to process large data, and improvement in deep learning models. Thus, the deep learning methods used in the study are briefly discussed.

1) CONVOLUTIONAL NEURAL NETWORK (CNN)

The convolutional artificial neural network (CNN) is one of the most important architectures of deep learning, which is a multi-layered feed-forward neural network model [24]. CNN extracts features according to the spatial principle. The two-pixel values in images related to each other enhance the classification performance of CNN on the images. Presently, this network structure is frequently used in text classification problems [4], [22].

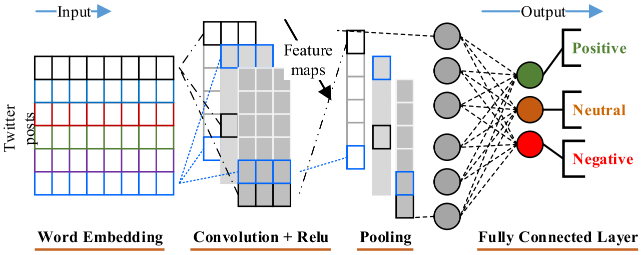


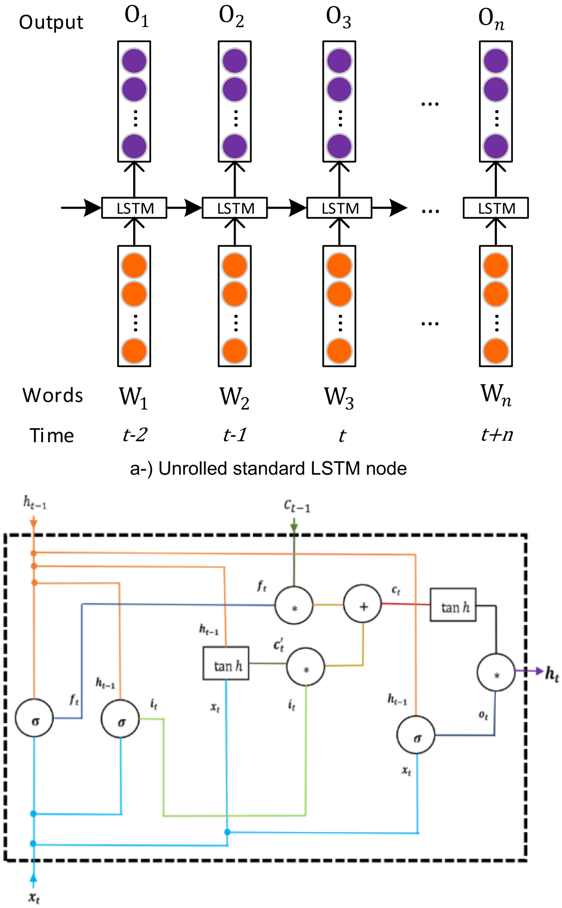
FIGURE 2. Stages of CNN architecture for text classification problems.

Basically, CNN consists of three layers: convolution, pooling, and fully connected layer. Unlike conventional artificial neural networks, a convolution layer automatically performs feature extraction and a pooling layer is used for feature reduction. In Figure 2, a CNN visual representation takes the text as an input and indicates the relationships between the layers of a CNN model to determine the class of the text. In the convolution layer, the feature is extracted from the image or text with the help of various filters. The intermediate process is applied between the convolution layer and the pooling layer so that the features are non-linear with the help of the Rectified Linear Unit activation function. In the pooling layer, the dimensions of these feature maps are reduced, which reduces the computational workforce in the subsequent layers and display the features in the image or text more effectively. The final layer of the convolutional neural network is in the form of a classical fully connected artificial neural network. In this layer, a fully connected structure between the artificial neurons represents the features of the image/text and the target class labels. The new text serves as an input to the CNN. When the training is completed, CNN gives the predictive class probability [5], [24].

2) LONG SHORT-TERM MEMORY (LSTM)

The RNN is a class of neural networks in which the outputs of the feed-forward classical artificial neural network are given as new input to the neurons based on new input values [5]. The output value at any neuron $(t + 1)$ depends on its input at the moment t. This adds dynamism to the network model. Assuming there is a relationship between two input values, this model is defined as a memory network model [25]. In RNN, input data is assumed to be related to each other. LSTM is also one of the most popular RNN network models, where the architecture is developed for vanishing gradient problem. Figure 3 (a) shows the unrolled LSTM in the time series. Here, w_t represents the input value at time t, and o_t represents the output value at time t.

Figure 3 (b) shows the architecture of the LSTM network node that consists of three basic gates such as the input gate i_t, the output gate o_t, and the forget gate f_t. Whereas the input gate and output gate represent the data entering and data leaving the node at time t, respectively. The forgetting gate decides the information be forgotten compared to the previous status information (h_{t-1}) and the current input (x_t). These three gates decide how to update the current memory cell c_t and the current latency h_t values. In an LSTM node, the relationships between the gates are calculated mathematically using the following equations [13]:



a-) Unrolled standard LSTM node

b-) Connections of gates of standard LSTM [24]

FIGURE 3. Architecture of the standard LSTM memory block.

$$\boldsymbol{i_t} = \sigma(w_i.[h_{t-1}, x_t] + b_i) \tag{1}$$

$$\boldsymbol{f_t} = \sigma(w_f.[h_{t-1}, x_t] + b_f) \tag{2}$$

$$\boldsymbol{o_t} = \sigma(w_o.[h_{t-1}, x_t] + b_o) \tag{3}$$

$$\boldsymbol{c_t'} = tanh(w_c.[h_{t-1}, x_t] + b_c) \tag{4}$$

$$\boldsymbol{c_t} = f_t * c_{t-1} + i_t * c_t{}' \tag{5}$$

$$\boldsymbol{h_t} = o_t * tanh(c_t) \tag{6}$$

The LSTM network architecture processes the representation vectors of a sentence taking as an input from the first word to the last word. This involves investigating the relationship between words from the beginning to the end. The BiLSTM network, used frequently today, can identify the long-term relationships between words from the beginning to the end and from the end to the beginning. The two-way input data processing creates extra workload calculation. In our study, LSTM and BiLSTM networks are used for the experiments of the proposed model.

TABLE 1. Confusion matrix for a binary classification problem.

		Predicted Values	
		Positive	*Negative*
Actual Values	*Positive*	**TP**	**FP**
	Negative	**FN**	**TN**

3) GATED RECURRENT UNIT (GRU)

The GRU is a widely used RNN network architecture. In RNN networks, the GRU was developed for the vanishing gradient problem, similar to the LSTM architecture [26]. However, this offers better performance than LSTM in many problems other than language modeling. GRU architecture is simpler than LSTM architecture. While LSTM architecture has input, output, and forget gate, GRU architecture has reset gate and update gate. Since GRU calculation is simpler than LSTM calculation, it performs faster calculations in addition to lower memory [5].

B. PERFORMANCE METRICS

The measurements obtained from the confusion matrix will be compared with the classification achievements obtained from sentiment classification in similar studies, to demonstrate the accuracy of the method. Accuracy, precision, and F1 measurement values are obtained from the confusion matrix. The simple confusion matrix for a two-class classifier is given in Table 1.

The abbreviations TP (True Positive), FP (False Positive), FN (False Negative) and TN (True Negative) in the confusion matrix in Table 1 have the following meanings:

- **TP:** Number of samples where the predicted class label is positive, and the actual class label is correct.
- **FP:** Number of instances where the predicted class label is positive, and the actual class label is incorrect.
- **FN:** Number of instances where the predicted class label is negative, and the actual class label is incorrect.
- **TN:** Number of samples where the predicted class label is negative, and the actual class label is correct.

Accuracy, Precision, Recall and F1 measurement are calculated according to the confusion matrix in Table 1. Calculation of the accuracy is made according to equation (7).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

Precision is the total estimate of the class labels accurately predicted for each class. The precision measure is calculated using equation (8).

$$Precision = \frac{TP}{TP + FP} \tag{8}$$

Recall value is the weighted average of the correct labels, correctly classified for each class. This value is calculated according to equation (9).

$$Recall = \frac{\mathrm{TP}}{TP + FN} \tag{9}$$

Other metrics, F1, is used to combine precision and recall values in a single measurement. The value of this measurement is between 0 and 1, and if the classifier correctly classifies all samples, it takes the value 1. F1 measure is given in equation (10), and the F1 value is close to 1 for a good classification success.

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{10}$$

IV. METHODOLOGY

In this section, we discuss the dataset, the data pre-processing, the structure of the proposed model, and the motivation in detail.

A. DATASET

The service quality offered by the Global System for Mobile Communications (GSM) operators is becoming more important daily. In this study, we used a dataset collected from shared user tweets about a GSM operator in Turkey. The dataset contains 17,289 Turkish tweets between 2011 and 2017. The tweets have three sentiment classes: positive, negative, and neutral. Class-based numbers of tweets in the dataset are given in Figure 4. Since the class distribution of the tweets is unequal, the dataset has an unbalanced distribution. The dataset consists of training and testing sets. Thus, we used training and testing sets in Figure 4 to compare the classification success of the proposed model with that of the previous studies that use the same dataset [27]. A total of 13,832 tweets was used for the training models, and 3,457 tweets were used for validation and testing models.

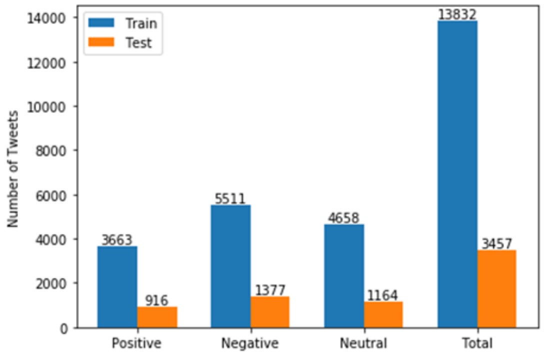


FIGURE 4. Class-based numbers of tweets in the dataset.

B. DATA PRE-PROCESSING

The text-based data produced by the users in the social media consist of a variety of content, except for alphabetic characters. However, these contents will not contribute to the intended work purpose. For example; “@username” will

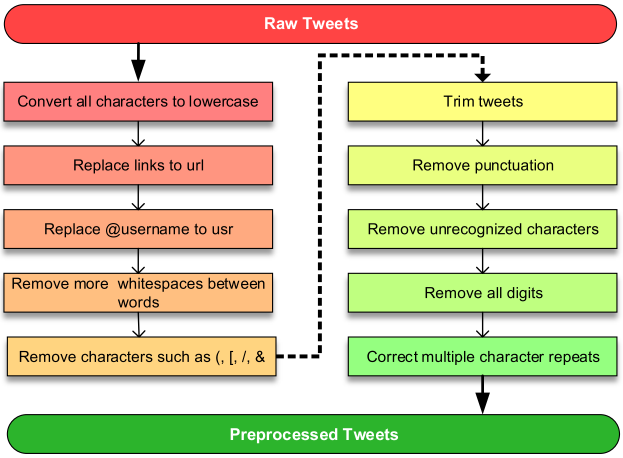

Raw Tweets
Convert all characters to lowercase
Replace links to url
Replace @username to usr
Remove more whitespaces between words
Remove characters such as (, [, /, &
Trim tweets
Remove punctuation
Remove unrecognized characters
Remove all digits
Correct multiple character repeats
Preprocessed Tweets

FIGURE 5. Data pre-processing steps used in this study.

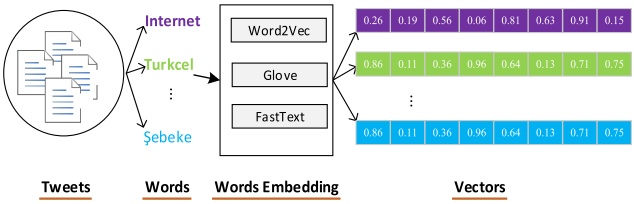


FIGURE 6. Basic word embedding stages for the mapping of words into numerical vectors.

not contribute positively or negatively to any post, aiming at conducting sentiment analysis on Twitter posts. These contents are called noise in text processing problems [28]. Based on the analysis from the previous studies, many algorithms increase the classification performance by cleaning textual data content [29], [30].

Since Twitter allows users to share messages of 280 characters, Twitter datasets often contain semantically compressed information. To achieve high classification success in classifying short and dense data, the text pre-processing steps in Figure 5 were applied to the dataset. The goal of these pre-processing is to reduce the noisy content to the normal form before the word-based representation phase. In this study, we did not perform any pre-processing on the dataset for the character-level representations, which was a branch of the proposed hybrid CNN model. In Figure 5, we performed pre-processes steps in the RNN-based classification using FastText and Word2Vec word representations. This represents the second branch of the proposed model. In the pre-processing steps, first, all characters in the dataset were converted to lowercase. The link information in the dataset is replaced with the "url" since the usernames in the dataset do not provide emotional content of the text, the usernames are replaced by "usr". In addition, the multiple spaces between words were reduced to one space. We also removed punctuation, numbers, and undefined characters in the tweets. Multiple character repeats, such as "turkcellllllll", frequent in tweets, have been corrected. We used Zemberek, which is the Turkish natural language processing (NLP) framework in the pre-processing phase [31]. We removed words written in other languages such as English and Ukrainian from the tweets with the help of Zemberek. In the Twitter ecosystem, users generally do not write the texts according to the correct grammar rules, and the contents contain many spelling mistakes. We used the Zemberek framework to correct these spelling mistakes, and we performed many experiments to evaluate the impact of pre-processing on our dataset. From the outcome of the experiments, it was observed that the pre-processing increased the classification performance. Thus, it is recommended to apply pre-processing for Turkish text classification problems.

C. PROPOSED HYBRID MODEL

In this section, we discuss the basic structure of the proposed model, motivation, experimental mechanisms, and approaches to test its effectiveness.

1) CORE IDEA

Text representation plays a critical role in many NLP tasks. Successful word embedding can facilitate text encoding and improve classification performance. With this approach, the dataset can be represented by different methods. The essence of this study is to increase the classification performance by combining the power of different word representations and different deep learning methods. In addition, CNN and LSTM provide effective performance on different data representations. While CNN has the ability to capture feature extraction in local regions, LSTM can extract good features from datasets with long-term dependencies such as natural languages and signals. This facilitates the successful application of CNN in datasets that have close semantic relationships like images. On the other hand, LSTM provides good performance on NLP problems and solves semantic dependence among the words. These methods are effective as they can contribute to the sentiment classification problems. This contribution has been confirmed by the obtained classification results. With word embedding methods such as Word2Vec and FastText, some contents in the text disappear. For example, contents such as URL information, emoji, stop words, etc. are removed during the pre-processing phase. However, these removed contents are part of the user's thought. To enhance the sentiment integrity, the combination of different text representations will express the integrity of the user opinions. In this study, the same tweets are represented by both FastText and character-level embedding. These different representations are the inputs of different deep learning methods and the extracted features. Figure 7 represents architecture of the proposed deep learning model. A branch of the proposed model works with the active CNN character-level embedding, whereas another branch works with the active LSTM FastText embedding. By combining the features obtained from both branches, the tweets are classified according to sentiment.

The dataset consisting of m tweets is denoted as $D = \{T_1, T_2, T_3, \ldots, T_m\}$, and m is equal to 17,289 total tweets in test and training dataset. Given a tweet T_i, the tweet

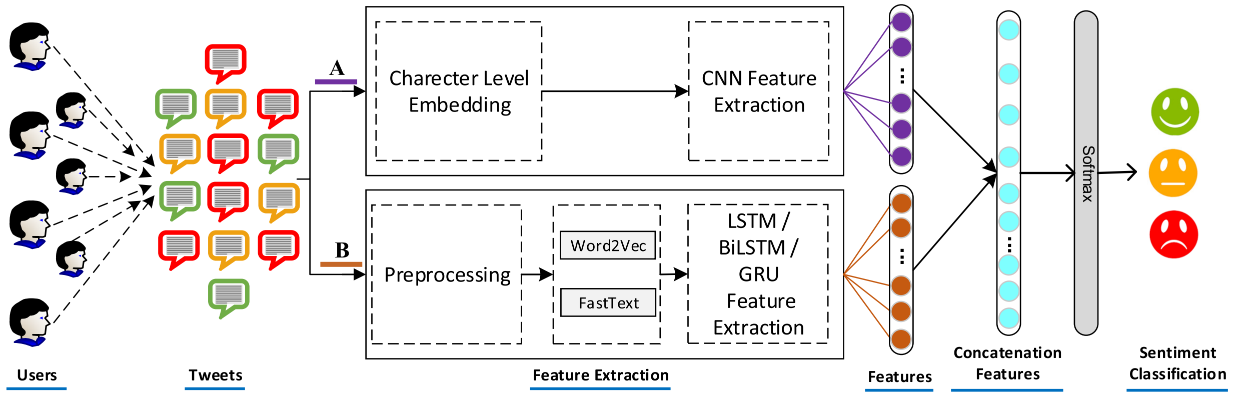


FIGURE 7. Architecture of the proposed hybrid deep learning model based on the strategic combination of different word embedding with different deep learning methods.

with k words is denoted as $T_i = \{w_{i,1}, w_{i,2}, w_{i,3}, \ldots, w_{i,k}\}$. In the branch B of the proposed model, we embed each word w_l to a pre-trained word vector where $w_l^{\omega} \in \mathbb{R}^d$ is the d-dimensional embedding vector of the lth word, and the word level embedding as $T_i^{\omega} = \{w_{i,1}^{\omega}, w_{i,2}^{\omega}, w_{i,3}^{\omega}, \ldots, w_{i,k}^{\omega}\}$. We give T_i^{ω} as input to BiLSTM for feature extraction, namely F_i^{BiLSTM} in equation 11.

$$F_i^{BiLSTM} = BiLSTM(T_i^{\omega}) \tag{11}$$

In the branch A of the proposed model, we assumed that V is vocabulary of characters and d is dimensionality of character-level embedding. The size of V is denoted as L. The character embedding matrix will be equal to $Q \in \mathbb{R}^{dxL}$. Moreover, while d is the height of Q embedding matrix, L is the length. The $w_{i,j}$, which is a word, consists of a sequence of characters $w_{i,j} = \{c_1, c_2, c_3, \ldots, c_p\}$. While p is the length of $w_{i,j}$, j is j-th word in T_i. The character-level embedding of the word $w_{i,j}$ is $C^{w_{i,j}} \in \mathbb{R}^{dxp}$. For each T_i, F_i^{CNN} features extracted with the help of CNN filter and Q matrix as stated in equation 12. Features obtained from both branches of the hybrid model are combined with the help of equation 13. Herein, $\oplus$ represents to concatenations operations.

$$F_i^{CNN} = CNN(T_i) \tag{12}$$

$$F^{Hybrid} = \sum_{i=1}^{m} F_i^{CNN} \oplus F_i^{BiLSTM} \tag{13}$$

The obtained F^{Hybrid} features are given as input to softmax layer of proposed hybrid model. In this way, high-level features are transmitted to the softmax layer. The main novelty between our hybrid model and the other hybrid models in the literature is that we combine different word embedding methods with different distinctive deep learning methods in order to obtain a better classification score.

2) TEXT REPRESENTATION METHODS

Computer algorithms can only work with numerical data. To process text-based data in a computer, these data must be represented in numerical format. The text representing the process is one of the critical NLP research. Methods such as Word2Vec [32], Glove [33], FastText [34], BoW [35] are the pioneer of word representation methods. In addition to these methods, texts can be represented numerically by introducing individual feature engineering to datasets [36].

Word representations are critical to many NLP tasks. Good word representations can better encode text and improve classification performance. Word2vec is a two-layer neural network that processes text and expresses words as vectors. This model, which takes data in text format as an input, produces a set of vectors as output. This set of vectors is feature vectors that represent words in the dataset. Glove, which is another common word embedding method, is an unsupervised learning algorithm used to obtain vectors from words. Glove represents the words according to the coexistence statistics of the words in the dataset. The main disadvantage of Word2Vec and Glove is their ability to generate a random vector in a word, not in the dataset. On the other hand, FastText, a continuation of Word2Vec, can overcome this disadvantage. The FastText uses the n-gram approach to create a word in the representation phase. This makes it better for embedding a word that is not in the corpus. The main disadvantage of this method is its more memory utilization during running. In this study, pre-trained Word2Vec and FastText word embedding methods were used to represent the dataset. Figure 6 shows the basic steps to convert texts to vector format.

Within the scope of this study, many experiments were performed to represent the dataset with Word2Vec and FastText methods. Since the representation method with FastText seems more successful, this embedding is used in the proposed model.

3) MODEL BASELINES

The proposed model can be compared to a tree structure with two branches. We recommend softmax, the root of the classification layer. On the other hand, branches are deep learning methods that extract features separately in different representations of data. With the help of CNN, a branch of this tree is used to extract features; another branch extract features with the help of LSTM. The features obtained from both branches that feed the root of the tree, and the fed tree root classifies tweets in terms of sentiments. Figure 7 represents the proposed deep learning model. In branch A, the features are extracted from the character-level embedding. Symmetrically in branch B, the features are extracted based on word representation methods. Since RNNs provide better performance on word embedding vectors [37], we use BiLSTM and word embedding together in the study. Following the same approach, CNN and character-level embedding used together [23].

The basic approach of the proposed model in branch A is to find out the effect of all the components (character, exclamation, number, emoji, abbreviation, etc.) in the tweets of the sentiment content. In other words, it is thought that every component in user sharing contributes to user sentiment. Therefore, it is desired to extract the features in terms of sentiment without pre-processing the dataset. When the dataset is investigated, while the neutral class tweets contain a lot of URLs, the positive and negative class tweets are found to contain less URL information. The URL distribution in the dataset is a feature of classification. While tweets are converted to character-level numerical vectors, word embedding is created based on all the unique characters in the context.

The main philosophy behind the proposed approach in branch B is to contribute to the tweet's classification in terms of sentiment by effectively revealing the relationships between the words in the tweets. In the proposed model, the dataset first goes through data pre-processing steps mentioned in the previous sections. In this way, the semantic relationship between words in the dataset is revealed. Later, the words in the tweet were converted into vectors using the current and effective Word2Vec and FastText word representation methods. Then, pre-trained Word2Vec and FastText word representation models were used on Turkish Wikipedia text documents. To determine the classification performance of the proposed model, the dataset was represented by both Word2Vec and FastText in each algorithm. Trained word representations are used because user messages are short and contain too many implications in accordance with the jargon of the Twitter. In the last stage of branch B, the features are extracted using RNN variants such as LSTM, BiLSTM, and GRU methods. In the last stage of the proposed model, the features extracted from A and B branches were combined and transmitted to the softmax layer for classification.

V. EXPERIMENTS AND RESULTS

Many experiments were performed to confirm the effectiveness of the proposed model. Two different experimental approaches are presented to evaluate the performance of the proposed hybrid model. The first approach is to compare the performance of the basic deep learning created through the classification success of the proposed model. In the second approach, the performance of the proposed model is compared with previous studies that focus on the importance of deep learning models on the text classification problems.

Many libraries and tools are available for developing deep learning models. Keras is one of the most preferred software frameworks [38]. Tensorflow is used in the backend of Keras [39], which provides support for the CPU or GPU environment. In this study, Keras and Tensorflow were run on the GPU. Deep learning experiments were performed on a computer with the following specifications: NVIDIA GeForce GTX TITAN Black 6 GB GPU, Intel Core i-7 processor, 24 GB RAM memory and SSD hard disk. Moreover, Google COLAB [40] was used for experiments when the computer hardware was insufficient. Each algorithm was run five times, and the highest average value was recorded.

In this study, we used the accuracy value as the main performance metric so that we would be able to compare our results with the results of the previous studies. In addition, we used different performance metrics such as F1, Kappa, Recall to evaluate the performance of our models.

A. BASIC MODELS

One of the most important steps in the text classification problem is to represent the texts correctly. In this study, the Turkish dataset shared by GSM operators is represented by character-level embedding, Word2Vec embedding, and FastText embedding. Thus, we have evaluated the performance of different representation methods against the same algorithm and parameters to investigate the performance of our proposed hybrid model and 12 basic deep learning models. These models are CNN and RNN based models, which are popular deep learning methods. The basic models are named as M-1, M-2, ..., M-12. Among these models, M-1 and M-7 have different characteristics. Since both M-1 and M-7 are branches of our hybrid model as seen in Figure 7, we used "M-1-A" and "M-7-B" notation, respectively. In addition, the proposed hybrid model is named M-Hybrid. The 12 basic models consisting of the four basic deep learning methods - CNN, LSTM, BiLSTM, and GRU - referred as feature extractors and classifiers. Many experiments were performed to tuning the parameters of the created models and to determine the most ideal activation and optimization functions. The best parameters for each basic model are determined in Table 2.

The most common 15,000 words were used in the corpus for the character-level, FastText, and Word2Vec embedding. Stochastic Gradient Descent (SGD) was used as the optimization function, and the learning rate of SGD parameters and momentum value were selected as 0.01 and 0.9, respectively. Dropout [41] method was used to prevent the overfitting of the models during training. The classification performance of the basic models is given in Table 3. The results of the FastText are better than that of the Word2Vec. It is seen that FastText and Word2Vec embedding, which are word representation methods, offer better results with RNNs models. When RNNs models are evaluated, while BiLSTM shows the best performance in FastText word embedding, Word2Vec word embedding offers the best performance in the GRU. Moreover, RNNs models seem to be very close to each other on the same word embedding methods.

TABLE 2. Parameters of basic deep learning models used in this study.

											Layers				
Model	**Classification Method**	**Embedding**	**Activation**	**Embedding Size**	**Dropout**	**Recurrent Dropout**	**Optimizers**	**Epochs**	**Filters**	**Kernel Size**	***Embedding***	***Conv1D***	***MaxPool1D***	***Dense***	***LSTM-BiLSTM-GRU***
M-1-A	CNN	Character	softmax	300	0.5	-	Sgd	50	512	3	1	1	1	3	-
M-2	LSTM	Character	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-3	BiLSTM	Character	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1(128 units)
M-4	GRU	Character	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-5	CNN	FastText	softmax	300	0.5	-	Sgd	50	512	3	1	1	1	3	-
M-6	LSTM	FastText	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-7-B	BiLSTM	FastText	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-8	GRU	FastText	softmax	300	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-9	CNN	Word2Vec	softmax	400	0.5	-	Sgd	50	512	3	1	1	1	3	-
M-10	LSTM	Word2Vec	softmax	400	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-11	BiLSTM	Word2Vec	softmax	400	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M-12	GRU	Word2Vec	softmax	400	0.5	0.4	Sgd	50	-	-	1	-	-	3	1 (128 units)
M- Hybrid	CNN + BiLSTM	Character + FastText	softmax	300	0.5	0.4	Sgd	50	512	3	1	1	1	3	1 (128 units)

The best accuracy from the word embedding approach is obtained by M-7-B, while the best accuracy from the character-level embedding is obtained by M-1-A. Therefore, we used M-1-A and M-7-B model in the M-Hybrid model together. The layer based architecture of the hybrid model is given in Figure 8. To compare the classification performance of the hybrid model with the performance of the M-1-A and M-7-B model, we develop a model consisting of two-input and three-output using the Keras functional application programming interface. The performances of the M-1-A and M-7-B models and the performances of the M-Hybrid model were observable at each iteration. In the models training phase, we performed experiments comprising of 20, 40, 50, 70, and 100 epochs. We achieved the best accuracy in 50 epochs. Therefore, we carried out all our experiments with 50 iterations.

During the test and validation of models, accuracy and loss curve of M-1-A, M-7-B, and M-Hybrid models over 50 iterations on the dataset are shown in Figure 9. It is evident from the Figure 9 (c) given above that the performance of the hybrid model was relatively affected by the M-1-A model during the training. As seen in Figure 9 (a) and (c), the classification accuracy of the proposed hybrid model was higher than the other basic models during most of the epochs. Moreover, it is clearly seen that the loss value of the M-Hybrid model is lower than the M-1-A and M-7-B in Figure 9 (b) and (d).

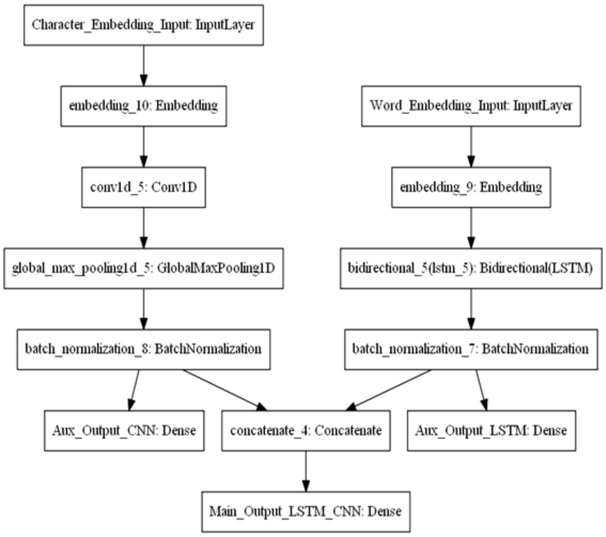
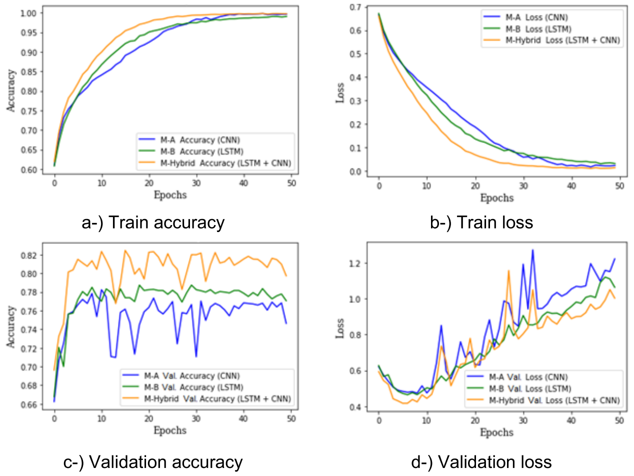

Character_Embedding_Input: InputLayer
embedding_10: Embedding
conv1d_5: Conv1D
global_max_pooling1d_5: GlobalMaxPooling1D
batch_normalization_8: BatchNormalization
Word_Embedding_Input: InputLayer
embedding_9: Embedding
bidirectional_5(lstm_5): Bidirectional(LSTM)
batch_normalization_7: BatchNormalization
Aux_Output_CNN: Dense
concatenate_4: Concatenate
Aux_Output_LSTM: Dense
Main_Output_LSTM_CNN: Dense

FIGURE 8. Layers of the M-Hybrid model used in this study.

Class-based classification metrics of three main deep learning models are given in Table 4. The classification accuracy of M-1-A, M-7-B, and M-Hybrid are 75.73%, 80.03%,

TABLE 3. Classification results of basic deep learning models based upon accuracy values.

Model	Classification Method	Embedding	Accuracy (%)
M-1-A	CNN	Character-level	**75.67**
M-2	LSTM	Character-level	73.71
M-3	Bi-LSTM	Character-level	73.11
M-4	GRU	Character-level	74.57
M-5	CNN	FastText	76.71
M-6	LSTM	FastText	79.41
M-7-B	Bi-LSTM	FastText	**80.44**
M-8	GRU	FastText	79.91
M-9	CNN	Word2Vec	78.06
M-10	LSTM	Word2Vec	79.07
M-11	Bi-LSTM	Word2Vec	79.18
M-12	GRU	Word2Vec	**79.59**
M-Hybrid	CNN + BiLSTM	Character + FastText	**82.14**



a-) Train accuracy

b-) Train loss

c-) Validation accuracy

d-) Validation loss

FIGURE 9. Accuracy and Loss curve of M-Hybrid, M-1-A, and M-7-B models during training and validation on the dataset.

and 81.77%, respectively. Since the dataset used in the study is unbalanced, metrics such as F1 and Kappa besides accuracy were also taken into account when evaluating the performance of the models. Class-based F1 and Recall score showed the highest performance in all three models for the negative class. In addition, the highest precision score was observed in the neutral class for all three models. Another important criterion to be considered in unbalanced datasets is the Cohen's kappa coefficient (K). This metric was used to measure inter-rater reliability for categorical instance's, and K varies from 0 to 1. The K value greater than 0,6 indicates substantial agreement between predicted and actual class. In our study, K values that were obtained as 0.62 for M-1-A, 0.69 for M-7-B, and 0.72 for M-Hybrid, respectively. Due to the fact that the K values we obtained are lower than the accuracy value shows that the models are affected by the unbalance of the dataset.

TABLE 4. Classification results of three main deep learning models based upon precision, recall, F1, and Kappa score.

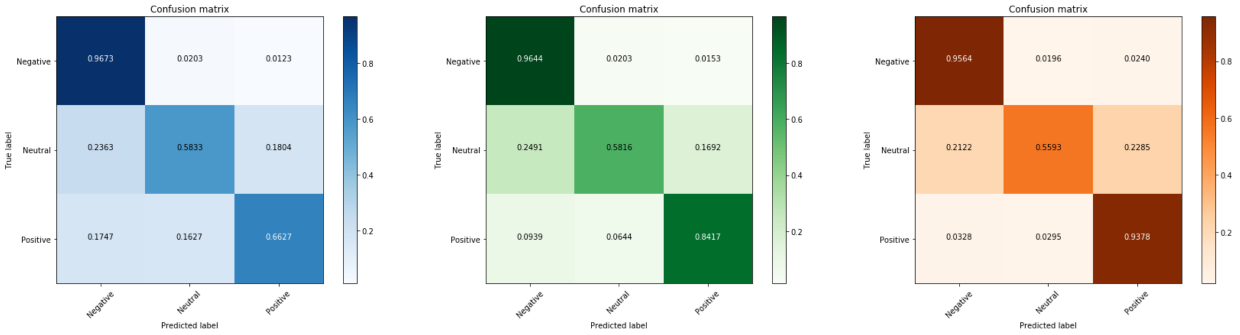
Model	Class	Precision	Recall	F1	Kappa (K)
M-1-A	*Negative*	0.75	**0.97**	**0.85**	0.62
	Neutral	**0.79**	0.58	0.67	
	Positive	0.73	0.66	0.69	
M-7-B	*Negative*	0.78	**0.96**	**0.86**	0.69
	Neutral	**0.89**	0.58	0.70	
	Positive	0.78	0.84	0.81	
M-Hybrid	*Negative*	0.83	**0.96**	**0.89**	0.72
	Neutral	**0.92**	0.56	0.70	
	Positive	0.74	0.94	0.83	

The confusion matrix of the three main deep learning models is given in Figure 10. During the test of three main models, we observed that the classification results of simple models and the proposed hybrid model can be distinct. In other words, while M-1-A models classified an input as neutral, the M-Hybrid model classified the same input as negative. For example, considering the tweet "*Turkcell'e kızgınım. Ve bu kızgınlık sanırım ayrılıkla sonlanıcak gibi geliyor bana. Farklı bir operatörün %30'u fazla fiyat teklif ediyorlar*" (i.e., *I am angry with Turkcell. And I think this anger seems to end with separation. They offer prices 30% more than a different operator*). Although the actual class of this tweet is negative, it classified as neutral by the M-1-A, negative by the M-7-B, and negative by M-Hybrid.

1) VISUALIZATION OF FEATURES IN MODELS

Understanding the data transformations that occur between layers of deep learning models is important for the improvement of better models. In our study, we used the t-Distributed Stochastic Neighbor Embedding (t–SNE) [42] method to see how data representations change between layers during the test of the M-1-A, M-7-B, and M-Hybrid model. We visualized feature maps for all three main models after embedding layer and before the softmax layer. We used three dimensions for visualizations of all features. The visualizations of features of the models are given in Figure 11. Moreover, negative, neutral, and positive class are represented by 0, 1, and 2, respectively. The effect of combining features obtained from two different data representation methods can be observed clearly in Figure 11 (e).

When we take a closer look at the features of M-1-A obtained before the softmax layer, it is possible to see that neutral and positive samples are located nearby. The main reason for this case is that sets (i.e. positive and negative instances) contain too many similar contents such as URL, # something, etc. When we examine the distribution of the data

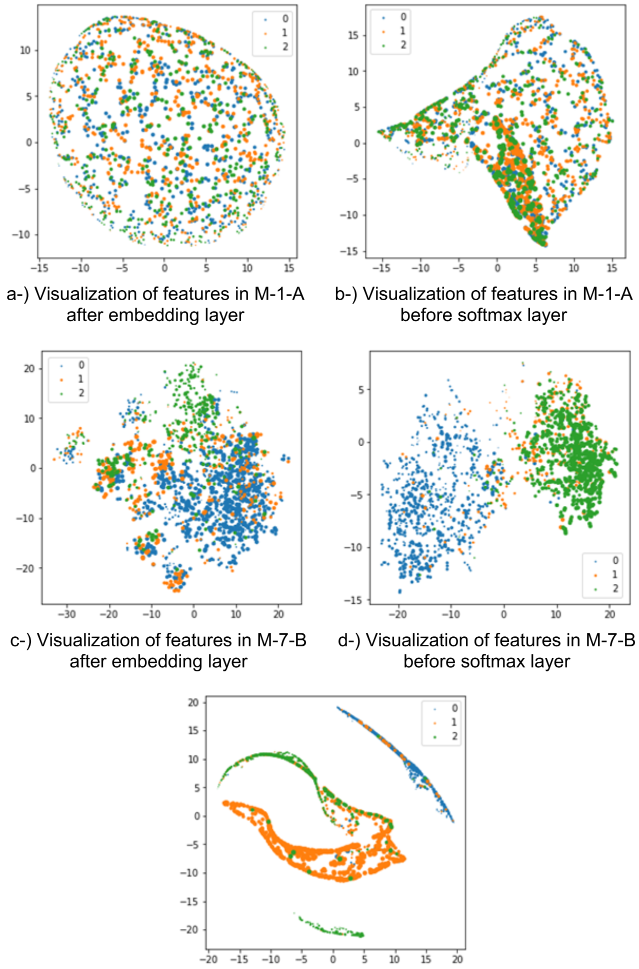


a-) Confusion matrix of M-1-A model

b-) Confusion matrix of M-7-B model

c-) Confusion matrix of M-Hybrid model

FIGURE 10. Confusion matrix of the three main deep learning models: a-) model M-1-A, b-) model M-7-B, and c-) model M-Hybrid respectively.



a-) Visualization of features in M-1-A after embedding layer

b-) Visualization of features in M-1-A before softmax layer

c-) Visualization of features in M-7-B after embedding layer

d-) Visualization of features in M-7-B before softmax layer

e-) Visualization of test dataset in M-Hybrid before softmax layer

FIGURE 11. Visualization of features in M-1-A, M-7-B, and M-Hybrid during the test of models.

TABLE 5. Wilcoxon rank sum test results of three main models.

Model pairs	P-value
M-1-A & M-7-B	0.001
M-1-A & M-Hybrid	0.001
M-7-B & M-Hybrid	0.003

TABLE 6. Comparison of classification results of baseline deep learning models with the proposed model based upon accuracy values.

Reference	Embedding	Method	Accuracy (%)
[15]	FastText Embedding	CNN	68.48
	Character Embedding	CNN	67.94
[44]	Character Embedding	CNN	60.57
[27]	FastText Embedding	LSTM	65.35
	Word Embedding	CNN	67.14
	Character Embedding	CNN	69.25
M-Hybrid	FastText Embedding + Character Embedding	CNN + BiLSTM	**82.14**

between layers in M-1-A and M-7-B, the features obtained by CNN are closer than those obtained by BiLSTM. As shown in Figure 11 (d) the negative samples are effectively separated from positive and neutral samples while others, that is, positive and neutral samples, are not separated well. In the analyses made with the naked eye on the dataset, it was seen that neutral and positive samples were very close to each other in terms of meaning. In addition, the distribution of the features obtained as a result of different embedding methods confirms that the performance of our proposed hybrid model is attractive.

2) STATISTICAL EVALUATION OF CLASSIFICATION PERFORMANCE

Statistical evaluation of the classification results of the proposed model is important for verifying the results of the model. In our study, we used the popular Wilcoxon Rank Sum test [43] so as to validate the results of our models. This test is used to determine whether any two independent samples come from populations with the same distribution. The metric of this test is *P*-value which used to evaluate the relationship or difference between two samples statistically. In general, the *P*-value of less than 0.05 represents a difference between the two datasets. In our study, each model was run 50 times and the classification results were used for the Wilcoxon Rank Sum test. The obtained *P*-values are shown in Table 5. The *P*-value less than 0.05 for all model pairs, it shows that there is an important relationship between the result of the models.

TABLE 7. Parameters of baseline deep learning models used in this study.

										Layers				
Reference	**Classification Method**	**Embedding**	**Embedding Size**	**Dropout**	**Recurrent Dropout**	**Optimizers**	**Epochs**	**Filters**	**Kernel Size**	***Embedding***	***Conv1D***	***MaxPool1D***	***Dense***	***LSTM-BiLSTM-GRU***
[15]	CNN	FastText	300	0.5	-	sgd	50	512	3-4-5	1	3	3	3	-
	CNN	Character	300	0.5	-	adam	50	512	10-7-5-3	1	4	4	3	-
[44]	CNN	Character	300	0.5	-	adam	50	512	7-3-3-3-3-3	1	6	6	3	-
[27]	LSTM	FastText	100	N/A	N/A	sgdm	30	-	-	1	-	-	3	N/A
	CNN	Word	100	0.66	-	sgdm	100	200	3	1	1	1	3	-
	CNN	Character	84	0.66	-	sgdm	200	200	3	1	1	1	3	-
M-Hybrid	CNN + BiLSTM	Character + FastText	300	0.5	0.4	sgd	50	512	3	1	1	1	3	1 (128 units)

B. COMPARISON WITH BASELINE MODELS

In this section, the implementations of the significant models run on the dataset. In addition, we compare our model results to another study using the same dataset. First, we implemented the Yoo Kim model that is based on character-level embedding and CNN [15]. Two sub-experiments were carried out for the Yoo Kim model: the character-level word embedding methods and CNN based model. In the first experiment, the dataset is an input to the model without pre-processing the dataset. Approximately 67% of the performance was achieved. In the second experiment, after the pre-processing of the dataset, FastText word vectors are given as an input to the model. Approximately 68% of performance was achieved from this experiment. We achieved similar accuracy in the two experiments.

In another experiment, the dataset was classified using the model developed by Zhang *et al.* [44]. In this experiment, approximately 60% of performance was achieved. Article implementation codes by [45] were used for both [15] and [44] models. The parameters of these models are given in Table 7.

In the last experiment, the performance of the proposed model was compared with the performances obtained from [27], only part of its dataset used in this study. The results of the three experiments are given in Table 6. The first results offer approximately 65% performance in in the classification based on usage LSTM and FastText together on the dataset. In the CNN classifications, the authors achieved nearly 67% accuracy with word-level embedding and 69% accuracy with character-level embedding. No information is provided about pre-processing and NLP steps in their study. In our model, these operations were also performed, and the importance of NLP sub-tasks and data pre-processing emerged in the classification of the Turkish datasets.

In the experiments, the performance of the proposed model was verified by running different text classification on deep learning. Since different representations in the proposed model make different levels of features be extracted from the dataset, the display of the dataset has been upgraded.

On the other hand, the proposed model has some limitations. At first, due to Twitter jargon, tweets contain a lot of sarcasm, implication, and special abbreviations. In our proposed model, we do not present any mechanism to handle the negative effects of these special usages in sentiment classification. Secondly, every word in tweets is not equally important in terms of sentiment. Indeed, adjectives and adverbs contain more significance than nouns in point of sentiment. Although emojis are good emotional indicators in a tweet, while we were creating text representations, we accepted them on par with other words. In our proposed model, we processed each word of the tweet with the same importance. Finally, deep learning models are more effective on huge datasets, but the dataset used in this study is not very large.

VI. CONCLUSION

A change in social media applications as a personal block and communication tool in the entertainment environment has increased the usage of these applications. User feedback on these applications has reached huge dimensions. Today, NLP and deep learning play a vital role in revealing the users' sentiments of these huge datasets. In this study, a new deep learning model is proposed to establish a strategic relationship between the representation of data in text format and deep learning methods. The proposed model is based on the principle that different deep learning models effective in different text representation methods.

In our model, we present a novel architecture that functions with character-level embedding and FastText embedding under CNN and LSTM algorithms. In one branch of proposed model, the features were extracted from the character-level embedding using CNN, whereas, in another branch, features were obtained from FastText embedding using BiLSTM. The features obtained from both branches were combined and transmitted to the softmax layer for classification.

Two different main experimental setups were carried out to verify the performance of the proposed model. In the first experiment, 12 basic deep learning models were created. Four different deep learning models (i.e. CNN, LSTM, BiLSTM, GRU) were used with three different text representation methods (i.e. Word2Vec, FastText, character-level embedding) together. In the experiment, FastText and Word2Vec embedding, which are word representation methods, offered better results with RNNs models. We achieved the best classification accuracy of 80.44% with combination BiLSTM and FastText in word embedding approach (i.e. namely M-7-B). On the other hand, we achieved the best classification accuracy of 75.67% with combination CNN and character-level representation (i.e. namely M-1-A). We achieved 82.14% classification success with our proposed M-Hybrid model, which combines extracted features from M-1-A and M-7-B models. From the result of the experiments, the performance of the proposed model is higher than that of other basic models.

In the second experiment, firstly, the performance of the proposed model was compared with the previous study on the same dataset. While the highest performance in previous study on the dataset was 69.25%, the performance of the M-Hybrid we proposed is 82.14%. Secondly, the dataset used in this study was classified with important deep learning methods in order to confirm the accuracy of our proposed model. The proposed model provided a higher classification performance than these important models.

In the light of the proposed method, we recommend the use of different text representation methods together for a better classification accuracy rate. This method would best suit to those of languages, such as Turkish, Arabic, and Lithuanian, which are difficult to analyze morphologically. In the future work, the proposed hybrid model can be improved enriched by attention mechanisms.

ACKNOWLEDGMENT

The authors would like to thank Assoc. Prof. Dr. M. F. Amasyali for sharing dataset used in this study.

REFERENCES

[1] L. Yue, W. Chen, X. Li, W. Zuo, and M. Yin, "A survey of sentiment analysis in social media," *Knowl. Inf. Syst.*, vol. 60, no. 2, pp. 617–663, 2019.

[2] F. Hemmatian and M. K. Sohrabi, "A survey on classification techniques for opinion mining and sentiment analysis," *Artif. Intell. Rev.*, vol. 52, no. 3, pp. 1495–1545, Oct. 2019.

[3] F. Zhu and X. Zhang, "Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics," *J. Marketing*, vol. 74, no. 2, pp. 133–148, Mar. 2010.

[4] A. U. Rehman, A. K. Malik, B. Raza, and W. Ali, "A hybrid CNN-LSTM model for improving accuracy of movie reviews sentiment analysis," *Multimedia Tools Appl.*, vol. 78, no. 18, pp. 26597–26613, Sep. 2019.

[5] F. Chollet, *Deep Learning With Python*. Shelter, Island: Manning, 2017.

[6] G. Liu, X. Xu, B. Deng, S. Chen, and L. Li, "A hybrid method for bilingual text sentiment classification based on deep learning," in *Proc. 17th IEEE/ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, May 2016, pp. 93–98.

[7] A. Yenter and A. Verma, "Deep CNN-LSTM with combined kernels from multiple branches for IMDb review sentiment analysis," in *Proc. IEEE 8th Annu. Ubiquitous Comput., Electron. Mobile Commun. Conf. (UEMCON)*, Oct. 2017, pp. 540–546.

[8] C. Zhou, C. Sun, Z. Liu, and F. C. M. Lau, "A C-LSTM neural network for text classification," 2015, *arXiv:1511.08630*. [Online]. Available: http://arxiv.org/abs/1511.08630

[9] Z. Z. Wint, Y. Manabe, and M. Aritsugi, "Deep learning based sentiment classification in social network services datasets," in *Proc. IEEE Int. Conf. Big Data, Cloud Comput., Data Sci. Eng. (BCD)*, Jul. 2018, pp. 91–96.

[10] S. Shi, M. Zhao, J. Guan, Y. Li, and H. Huang, "A hierarchical lstm model with multiple features for sentiment analysis of sina weibo texts," in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Dec. 2017, pp. 379–382.

[11] J. Kapočiūtė-Dzikienė, R. Damaševičius, and M. Woźniak, "Sentiment analysis of lithuanian texts using traditional and deep learning approaches," *Computers*, vol. 8, no. 1, p. 4, 2019.

[12] A. Shoukry and A. Rafea, "A hybrid approach for sentiment classification of egyptian dialect tweets," in *Proc. 1st Int. Conf. Arabic Comput. Linguistics (ACLing)*, Apr. 2015, pp. 78–85.

[13] Y. Guo, W. Li, C. Jin, Y. Duan, and S. Wu, "An integrated neural model for sentence classification," in *Proc. Chin. Control Decis. Conf. (CCDC)*, Jun. 2018, pp. 6268–6273.

[14] S. Hashida, K. Tamura, and T. Sakai, "Classifying sightseeing tweets using convolutional neural networks with multi-channel distributed representation," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2018, pp. 178–183.

[15] Y. Kim, Y. Jernite, D. Sontag, and M. A. Rush, "Character-aware neural language models," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2741–2749.

[16] K. Zhou and F. Long, "Sentiment analysis of text based on CNN and bi-directional LSTM model," in *Proc. 24th Int. Conf. Autom. Comput. (ICAC)*, Sep. 2018, pp. 1–5.

[17] B. Sun, F. Tian, and L. Liang, "Tibetan micro-blog sentiment analysis based on mixed deep learning," in *Proc. Int. Conf. Audio, Lang. Image Process. (ICALIP)*, Jul. 2018, pp. 109–112.

[18] J. Zheng and L. Zheng, "A hybrid bidirectional recurrent convolutional neural network attention-based model for text classification," *IEEE Access*, vol. 7, pp. 106673–106685, 2019.

[19] P. Kaladevi and K. Thyagarajah, "Integrated CNN- and LSTM-DNN-based sentiment analysis over big social data for opinion mining," *Behav. Inf. Technol.*, pp. 1–9, Dec. 2019. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/0144929x.2019.1699960, doi: 10.1080/0144929X.2019.1699960.

[20] G. Xu, Y. Meng, X. Qiu, Z. Yu, and X. Wu, "Sentiment analysis of comment texts based on BiLSTM," *IEEE Access*, vol. 7, pp. 51522–51532, 2019.

[21] A. Feizollah, S. Ainin, N. B. Anuar, N. A. B. Abdullah, and M. Hazim, "Halal products on Twitter: Data extraction and sentiment analysis using stack of deep learning algorithms," *IEEE Access*, vol. 7, pp. 83354–83362, 2019.

[22] Z. Zhang, Y. Zou, and C. Gan, "Textual sentiment analysis via three different attention convolutional neural networks and cross-modality consistent regression," *Neurocomputing*, vol. 275, pp. 1407–1415, Jan. 2018.

[23] Y. Xiao and K. Cho, "Efficient character-level document classification by combining convolution and recurrent layers," 2016, *arXiv:1602.00367*. [Online]. Available: http://arxiv.org/abs/1602.00367

[24] A. Yadav and D. K. Vishwakarma, "Sentiment analysis using deep learning architectures: A review," *Artif. Intell. Rev.*, pp. 1–51, Dec. 2019. [Online]. Available: https://link.springer.com/article/10.1007/s10462-019-09794-5, doi: 10.1007/s10462-019-09794-5.

[25] R. C. Staudemeyer and E. Rothstein Morris, "Understanding LSTM - a tutorial into long short-term memory recurrent neural networks," 2019, *arXiv:1909.09586*. [Online]. Available: http://arxiv.org/abs/1909.09586

[26] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN Encoder–Decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1724–1734.

[27] M. F. Amasyali, H. Tasköprü, and K. Çaliskan, "Words, meanings, characters in sentiment analysis," in *Proc. Innov. Intell. Syst. Appl. Conf. (ASYU)*, Oct. 2018, pp. 1–6.

[28] A. Sarker, "A customizable pipeline for social media text normalization," *Social Netw. Anal. Mining*, vol. 7, no. 1, pp. 1–13, Dec. 2017.

[29] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," *Inf. Process. Manage.*, vol. 50, no. 1, pp. 104–112, Jan. 2014.

[30] M. U. Salur and I. Aydin, "The impact of preprocessing on classification performance in convolutional neural networks for turkish text," in *Proc. Int. Conf. Artif. Intell. Data Process. (IDAP)*, Sep. 2018, pp. 1–4.

[31] A. A. Akin and M. D. Zemberek, "Zemberek, an open source NLP framework for turkic languages," *Structure*, vol. 10, pp. 1–5, Dec. 2007.

[32] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[33] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[34] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Trans. Assoc. Comput. Linguistics*, vol. 5, pp. 135–146, Dec. 2017.
[35] T. K. Landauer and S. T. Dumaıs, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.," *Psychol. Rev.*, vol. 104, no. 2, pp. 211–240, 1997.
[36] İ. Aydın, M. U. SALUR, and B. Fatma, "Multiple population based particle swarm optimization for sentiment analysis," *Türkiye Bilişim Vakfı Bilgi. Bilim. Mühendisliği Derg.*, vol. 11, no. 1, pp. 52–64, 2018.
[37] S. Seo, C. Kim, H. Kim, K. Mo, and P. Kang, "Comparative study of deep learning-based sentiment classification," *IEEE Access*, vol. 8, pp. 6861–6875, 2020.
[38] Chollet François. (2015). *Keras: The Python Deep Learning library*. Accessed: Jan. 10, 2020. [Online]. Available: https://keras.io/
[39] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: https://arxiv.org/abs/1603.04467
[40] Google. *Google Colab*. Accessed: Jan. 10, 2020. [Online]. Available: https://colab.research.google.com
[41] J. Xiong, K. Zhang, and H. Zhang, "A vibrating mechanism to prevent neural networks from overfitting," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2019, pp. 1929–1958.
[42] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.
[43] F. Wilcoxon, "Individual comparisons by ranking methods author," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.
[44] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 557–649.
[45] C. Joshi. (2018). *Character Level CNN*. Accessed: Jan. 15, 2020. [Online]. Available: https://github.com/chaitjo/character-level-cnn

MEHMET UMUT SALUR received the B.Sc. and M.Sc. degrees in computer engineering from Pamukkale University, Denizli, Turkey, in 2013, and 2016, respectively. He is currently pursuing the Ph.D. degree in computer engineering with Fırat University, Elazığ, Turkey. He is also a Research Assistant with Harran University. His research interests include machine learning, deep learning, natural language processing, and sentiment analysis.

ILHAN AYDIN was born in Elazığ, Turkey, in 1981. He received the B.Sc. and M.Sc. degrees in computer engineering and the Ph.D. degree in electrical and electronics engineering from Fırat University, Elazığ, in 2001, 2006, and 2011, respectively. He is currently an Associate Professor of computer engineering with Fırat University. His research interests include soft computing, optimization, real-time systems, sentiment analysis, and deep learning.