applied sciences

MDPI

Article

Sentiment Classification Using Convolutional Neural Networks

Hannah Kim and Young-Seob Jeong *

Department of Future Convergence Technology, Soonchunhyang University, Asan-si 31538, Korea; hannah@sch.ac.kr

* Correspondence: bytecell@sch.ac.kr

Received: 29 April 2019; Accepted: 22 May 2019; Published: 7 June 2019

Abstract: As the number of textual data is exponentially increasing, it becomes more important to develop models to analyze the text data automatically. The texts may contain various labels such as gender, age, country, sentiment, and so forth. Using such labels may bring benefits to some industrial fields, so many studies of text classification have appeared. Recently, the Convolutional Neural Network (CNN) has been adopted for the task of text classification and has shown quite successful results. In this paper, we propose convolutional neural networks for the task of sentiment classification. Through experiments with three well-known datasets, we show that employing consecutive convolutional layers is effective for relatively longer texts, and our networks are better than other state-of-the-art deep learning models.

Keywords: deep learning; convolutional neural network; sentiment classification

1. Introduction

In the Big Data era, the amount of various data, such as image, video, sound, and text, is increasing exponentially. As text is the largest among them, studies related to text analysis have been actively conducted from the past to the present. In particular, text classification has drawn much attention because the text may have categorical labels such as sentiment (e.g., positive or negative), author gender, language category, or various types (e.g., spam or ham). For example, the users of Social Network Services (SNS) mostly represent their sentimental feeling, and they often share some opinions about daily news with the public or friends. Emotional analysis involves mood categories (e.g., happiness, joy, satisfaction, angry), while sentiment analysis involves categories such as positive, neutral, and negative. In this paper, we target the sentiment analysis that classifies the given text into one of sentiment categories. On websites about movies, people are likely to post their comments that probably contain sentiment or opinions. If such a sentiment is accurately predicted, then it will be applicable to various industrial fields (e.g., movie recommendation, personalized news-feed). Indeed, the international market of movies is growing much faster than before, so many companies (e.g., Netflix) provide movie recommendation services that essentially predict the sentiment or rating scores of customers.

There have been many studies that have adopted machine learning techniques for text classification. Although the machine learning techniques have been widely used and have shown quite successful performance, they strongly depend on manually-defined features, where the feature definition requires much effort of domain experts. For this reason, deep learning techniques have been drawing attention recently, as they may reduce the effort for the feature definition and achieve relatively high performance (e.g., accuracy). In this paper, we aim at sentiment classification for text data and propose an architecture of the Convolutional Neural Network (CNN) [1–3], which is a type of

deep learning model. We demonstrate the effectiveness of our proposed network through experimental comparison with other machine learning models.

The contributions of this paper can be summarized as follows: (1) we design an architecture of two consecutive convolutional layers to improve performance for long and complex texts; (2) we provide a discussion about the architectural comparison between our models and two other state-of-the-art models; (3) we apply the CNN model to binary sentiment classification and achieved 80.96%, 81.4%, and 70.2% weighted F1 scores for Movie Review (MR) data [4–6], Customer Review (CR) data [7], and Stanford Sentiment Treebank (SST) data [8], respectively; and (4) we also show that our model achieved 68.31% for ternary sentiment classification with the MR data.

The remainder of this paper is organized as follows. Section 2 reviews related studies about sentiment classification, machine learning, and deep learning models. Section 3 presents a detailed description of our proposed network. Section 4 describes the experimental settings and datasets and compares the proposed model with some other models. Thereafter, Section 5 discusses the experimental results. Finally, Section 6 concludes this paper.

2. Background

2.1. Machine Learning for Sentiment Classification

The sentiment can be defined as a view of or an attitude toward a situation or event. It often involves several types of self-indulgent feelings: happiness, tenderness, sadness, or nostalgia. One may define the sentiment labels as a polarity or valence (e.g., positive, neutral, and negative) or several types of emotional feeling (e.g., angry, happy, sad, proud). The definition of sentiment labels will affect the outcome of the sentiment analysis, so we need to define the sentiment labels carefully. There have been studies that defined three or more sentiment labels (e.g., opinion rating scores, emotional feelings) [9–13], and some studies adopted two-dimensional labels (e.g., positive and negative) [14–26]. Although there has been much performance improvement in the field of sentiment analysis, the binary classification of sentiment still remains challenging; for example, the performance (e.g., accuracy) of recent studies varied from 70%–90% in terms of the characteristics of data. In this paper, we aim at the binary classification (positive or negative) and the ternary classification (positive, neutral, and negative).

There have been many studies on classifying sentiments using machine learning models, such as Support Vector Machine (SVM), Naive Bayes (NB), Maximum Entropy (ME), Stochastic Gradient Descent (SGD), and ensemble. The most widely-used features for such machine learning models have been n-grams. Read [14] used unigram features for sentiment binary classification and obtained 88.94% accuracy using the SVM. Kennedy and Inkpen [15] adopted unigram and bigram features for sentiment binary classification and achieved 84.4% accuracy for movie review data [27] using the SVM. Wan [20] tackled binary classification with unigram and bigram features and achieved 86.1% accuracy for translated Amazon product review data. In [21], Akaichi utilized a combination of unigram, bigram, and trigram features and obtained 72.78% accuracy using the SVM. Valakunde and Patwardhan [28] aimed at five-class (e.g., strong positive, positive, neutral, negative, and strong negative) sentiment classification and obtained 81% accuracy using the SVM with bigram features. In the study of Gautam and Yadav [18], they utilized the SVM along with the semantic analysis model for the sentiment binary classification of Twitter texts and achieved 89.9% accuracy using unigram features. Tripathy et al. [19] also used the SVM with n-gram features and obtained 88.94% accuracy for sentiment binary classification. Hasan et al. [25] adopted unigram features for sentiment binary classification and achieved 79% using the NB for translated Urdu tweets data. All of these studies that utilized n-gram features generally achieved 70–90% accuracies, and the most effective model was the SVM.

There also have been studies that have defined hand-crafted features for sentiment classification. Yassine and Hajj [29] used affective lexicon, misspelling, and emoticons as features and obtained

87% accuracy for ternary classification using the SVM. Denecke [22] defined three scores (e.g., positivity, negativity, and objectivity) as features and achieved 67% precision and 66% recall for binary classification using the Logistic Regression (LR) classifier. Jiang et al. [16] tackled binary classification using the SVM and achieved 67.8% accuracy for Twitter texts, where they adopted two kinds of target-independent features (e.g., twitter content features and sentiment lexicon features). Bahrainian and Dengel [23] used the number of positive words and the number of negative words as features and achieved 86.7% accuracy for binary classification with Twitter texts. Neethu and Rajasree [17] combined unigram features and their own Twitter-specific features and obtained 90% accuracy for binary classification using the SVM. Karamibekr and Ghorbani [30] defined the number opinion nouns as a feature and combined the feature with unigrams. They achieved 65.46% accuracy for ternary classification using the SVM. Antai [24] used the normalized frequencies of words as features and obtained 84% accuracy for binary classification using the SVM. Ghiassi and Lee [31] aimed at five-class sentiment classification, and they defined a domain-independent feature set for Twitter data. They achieved a 92.7% F1 score using the SVM. Mensikova and Mattmann [26] utilized the results of Named Entity (NE) extraction as features and obtained a 0.9 False Positive Rate (FPR). These studies elaborated on feature definition rather than using only n-gram features, and they accomplished better performance (e.g., 92.7% F1 score). It is not fair, of course, to compare the performance between the previous studies because they used different datasets. However, as shown in [17,30], it is obvious that the combination of hand-crafted features and the n-grams must be better than using only n-gram features.

Although there has been success using machine learning models with the hand-crafted features and the n-gram features, these studies have a common limitation that their performance varied depending how well the features were defined; for different data, much effort of domain experts will be required to achieve better performance. This limitation also exists in information fusion approaches for sentiment analysis [32] that combine other resources (e.g., ontology, lexicon), as it will cost much time and effort of domain experts. Deep learning model is one of the solutions for such a limitation, as it is known to capture arbitrary patterns (i.e., features) automatically. Furthermore, as stated in [33], using the deep learning model for sentiment analysis will provide a meta-level feature representation that generalizes well on new domains.

In this paper, we take the deep learning technique to tackle the sentiment classification. In the next subsection, we review previous studies that adopted the deep learning techniques for the sentiment classification.

2.2. Deep Learning for Sentiment Classification

The deep learning technique, one of the machine learning techniques, has been recently widely used to classify sentiments. Dong et al. [34] designed a new model, namely the Adaptive Recursive Neural Network (AdaRNN), that classifies the Twitter texts into three sentiment labels: positive, neutral, and negative. By experimental results, the AdaRNN achieved 66.3% accuracy. Huang et al. [35] proposed Hierarchical Long Short-Term Memory (HLSTM) and obtained 64.1% accuracy on Weibo tweet texts. Tang et al. [36] introduced a new variant of the RNN model, the Gated Recurrent Neural Network (GRNN), which achieved 66.6% (Yelp 2013–2015 data) and 45.3% (IMDB data) accuracies. All of these studies commonly assumed that there were three or more sentiment labels.

Meanwhile, Qian et al. [37] utilized Long Short-Term Memory (LSTM) [38–41] for binary classification of sentiment and obtained 82.1% accuracy on the movie review data [27]. Kim [1] had a result of a maximum of 89.6% accuracy with seven different types of data through their CNN model with one convolutional layer. Zhang et al. [42] proposed three-way classification and obtained a maximum 94.2% accuracy with four datasets, where their best three-way model was NB-SVM. Severyn and Moschitti [43] employed a pretrained Word2Vec for their CNN model and achieved 84.79% (phrase-level) and 64.59% (message-level) accuracies with SemEval-2015 data. The CNN model used in [43] was essentially the same as the model of [1]. Deriu et al. [44] trained the CNN model that

had a combination of two convolutional layers and two pooling layers to classify tweet data of four languages sentimentally and obtained a 67.79% F1 score. In the study of Ouyang et al. [45], the CNN model, which had three convolution/pooling layer pairs, was proposed, and the model outperformed other previous models including the Matrix-Vector Recursive Neural Network (MV-RNN) [46]. We conducted experiments with several of our CNN models that had different structures, and two of our models (e.g., the seventh and eighth models) were similar to the model of [44,45].

As summarized above, for the sentiment classification, there are two dominant types of deep learning technique: RNN and CNN. In this work, we propose a CNN model, the structure of which was carefully designed for effective sentiment classification. In the next subsection, we explain the advantages of the CNN in analyzing text.

2.3. Convolutional Neural Network for Text Classification

Among the existing studies using deep learning to classify texts, the CNN takes advantage of the so-called convolutional filters that automatically learn features suitable for the given task. For example, if we use the CNN for the sentiment classification, the convolutional filters may capture inherent syntactic and semantic features of sentimental expressions, as shown in [47]. It has been shown that a single convolutional layer, a combination of convolutional filters, might achieve comparable performance even without any special hyperparameter adjustment [1]. Furthermore, the CNN does not require expert knowledge about the linguistic structure of a target language [48]. Thanks to these advantages, the CNN has been successfully applied to various text analyses: semantic parsing [49], search by query [50], sentence modeling [2].

One may argue that the Recurrent Neural Network (RNN) [51] might be better for the text classification than for the CNN, as it preserves the order of the word sequence. However, the CNN is also capable of capturing sequential patterns, as concerns the local patterns by the convolutional filters; for example, the convolutional filters along with the attention technique have been successfully applied to machine translation [52]. Moreover, compared to the RNN, the CNN mostly has a smaller number of parameters, so that the CNN is trainable with a small amount of data [43]. The CNN is also known to explore the richness of pretrained word embeddings [53].

In this paper, we design a CNN model for the sentiment classification and show that our network is better than other deep learning models through experimental results.

3. The Proposed Method

CNN, which has been widely used on image datasets, extracts the significant features of the image, as the "convolutional" filter (i.e., kernel) moves through the image. If the input data are given as one-dimensional, the same function of CNN could be used in the text as well. In the text area, while the filter moves, local information of texts is stored, and important features are extracted. Therefore, using CNN for text classification is effective.

Figure 1 shows a graphical representation of the proposed network. The network consisted of an embedding layer, two convolutional layers, a pooling layer, and a fully-connected layer. We padded the sentence vectors to make a fixed size. That is, too long sentences were cut to a certain length, and too short sentences were appended with the [PAD] token. We set the fixed length S to be the maximum length of the sentences. An embedding layer that maps each word of a sentence to an E-dimensional feature vector outputs an $S \times E$ matrix, where E denotes the embedding size. For example, suppose that 10 is king, 11 is shoes, and 20 is queen in the embedding space. 10 and 20 are close in this space due to the semantic similarity of king and queen, but 10 and 11 are quite far because of the semantic dissimilarity of king and shoes. In this example, 10, 11, and 20 are not numeric values, they are just the simple position in this space. In other words, the embedding layer is a process of placing words received as input into a semantically well-designed space, where words with similar meanings are located close and words with opposite meanings are located far apart, digitizing them into a vector. The embedding is the process of projecting a two-dimensional matrix into a low-dimensional vector

space (E-dimension) to obtain a word vector. The embedding vectors can be obtained from other resources (e.g., Word2Vec) or from the training process. In this paper, our embedding layer was obtained through the training process, and all word tokens including the [UNK] token for unseen words would be converted to numeric values using the embedding layer.

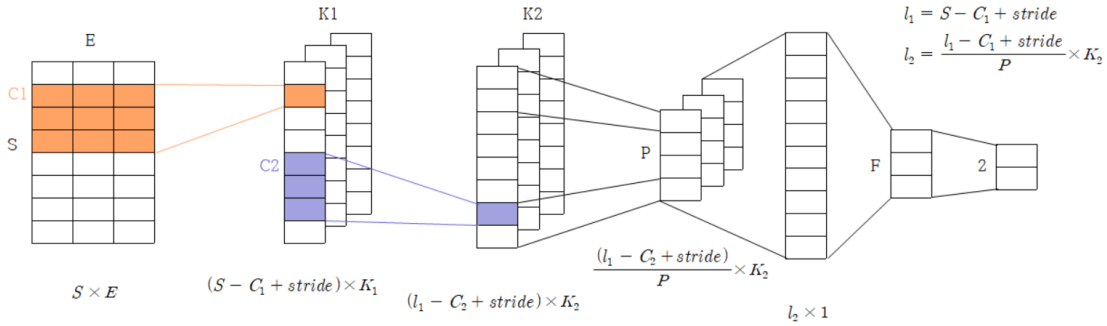

Figure 1. The graphical representation of the network, where the output dimensions of each layer are represented at the bottom of the corresponding layers.

The $S \times E$ matrix, the output of the embedding layer, is then laid down as the first convolutional layer. The first convolutional layer is the $C_1 \times E$ matrix, which stores the local information needed to classify the sentiment class in a $S \times E$ matrix and convey information to the next convolutional layer. The $C_1 \times E$ matrix slides (i.e., convolves) all the values of the $S \times E$ matrix with an arbitrary stride, calculates the dot product, and passes the dot product result to the next layer. The second convolutional layer uses the $C_2 \times 1$ matrix to extract features from the contextual information of the main word based on the local information stored in the first convolutional layer. C_1 and C_2 denote the filter size of each convolutional layer, and the two convolutional layers have K_1 and K_2 distinct filters, respectively, to capture unique contextual information. In other words, the first convolutional layer is utilized to look at simple contextual information while looking over the $S \times E$ matrix, and the second convolutional layer is utilized to capture key features and then extract them (e.g., worst, great) that contain sentiments affecting classification.

The matrix that passed through the consecutive convolutional layer is used as the input to the pooling layer. While average-pooling and L2-norm pooling have been used as the pooling layer position, in this paper, we used the max-pooling, which is a technique for selecting the largest value as a representative of the peripheral values. Since the sentiment is often determined by a combination of several words rather than expressing the sentiment in every word in the sentence, we adopted the max-pooling technique. The pooling layer slides all the values of the matrix, which is the output of the second convolutional layer, with an arbitrary stride, resulting in output vectors. Since max-pooling is the layer that passes to the next layer the largest value among several values, it results in output vectors of a much smaller size. In other words, the convolutional layer looks at the context and extracts the main features, and the pooling layer plays a role in selecting the most prominent features.

After passing through the pooling layer, a flattening process is performed to convert the two-dimensional feature map from the output into a one-dimensional format and to delivery it to an F-dimensional Fully-Connected (FC) layer. Since the FC layer takes a one-dimensional vector as the input, the two-dimensional vector delivered from the pooling layer needs to be flattened. The FC layer connects all input and output neurons. A vector that passes through the FC layer forms an output that is classified as positive or negative. The activation function softmax functions to classify multiple classes in the FC layer. The softmax function outputs, the value, which is the probability value, is generated for each class.

Most people might think that with many convolutional layer stacks, it may be better to store local information and to extract contextual information; however, deep networks do not always have higher performance than shallow networks. Because a network's variation (e.g., deep or shallow) might rely

on the length of the data and the number of data and features, in this paper, we argue that data passing through two consecutive convolutional layers and then passing through the pooling layer is successful at storing context information and extracting prominent features. This assertion is demonstrated by experiments and discussed in Section 5.

4. Experiment

4.1. Data

We used three Movie Review (MR) data from kaggle [4–6]. The dataset in [4] consisted of two labels, positive and negative, while [5] was composed of three labels of positive, neutral, and negative. Furthermore, the dataset in [6] was composed of five labels of positive, somewhat positive, neutral, somewhat negative, and negative. The positive class of [4], the positive class of [5], and the positive and somewhat positive classes of [6] were merged and labeled as the positive class. The neutral class of [5] and the neutral class of [6] were merged and labeled as the neutral class. Then, the negative class of [4], the negative class of [5], and the somewhat negative and negative classes of [6] were merged and labeled as the negative class. Positive, neutral, and negative were assigned as 1, 0, and 2, respectively. Some 11,800 positive data, 5937 neutral data, and 9698 negative data were used, totaling 27,435. Amazon Customer Review data (CR) [7] were labeled as positive and negative, and the amount of positive data was more than that of negative data. Therefore, we used only 3671 out of 34,062 data to control the positive:negative ratio. Stanford Sentiment Treebank data (SST) [8] were labeled as a value between 0 and 1, so a value of 0.5 or more was relabeled as positive, and a value of less than 0.5 was relabeled as negative.

Table 1 shows the detailed statistics of the data. N is the number of text data, Dist (+, −) is the ratio of positive:negative classes, and aveL/maxL is the average length/maximum length of the text. We divide all the data into three sets (i.e., train, validation, and test) with a ratio of 55:20:25, and $|V|$ was the dictionary size of each dataset.

Table 1. Detailed statistics of the data.

Data	N	Dist (+,−)	aveL/maxL	Train:Test:Val	$\|V\|$
MR	21,498	55:45	31/290	12,095:5375:4031	9396
CR	3671	62:38	19/227	2064:918:689	1417
SST	11,286	52:48	12/41	6348:2822:2116	3550

4.2. Preprocessing

Preprocessing was carried out to modify the text data appropriately in the experiment. We used decapitalization and did not mark the start and end of the sentences. We deleted #, two or more spaces, tabs, Retweets (RT), and stop words. We also changed the text that represented the url that began with “http” to [URL] and the text that represented the account ID that began with “@” to [NAME]. In addition, we changed digits to [NUM], and special characters to [SPE]. We changed “can’t” and “isn’t” to “can not” and “is not”, respectively, since “not” is important in sentiment analysis.

We split the text data by space and constructed a new dictionary using only the words appearing more than six times in the whole text. In the MR data, 9394 words out of a total of 38,195 words were made into a dictionary. In CR data, 1415 out of a total of 5356 words, and in SST data, 3548 out of a total of 16,870 words were made. The new dictionary also included [PAD] for padding and [UNK] to cover missing words. Padding is a method of cropping all text to a fixed length and filling it with the [PAD] token for text shorter than that length. Here, the fixed length was the maximum length of text in each dataset.

4.3. Performance Comparison

Decision Tree (DT), Naive Bayes (NB), Support Vector Machine (SVM) [54], and Random Forest (RF) [55] were used for comparing with our CNN models. Table 2 summarizes the description of the traditional machine learning models and the parameter settings. The parameters were optimized via a grid searching. We also compared our CNN models with the state-of-the-art models such as Kim's [1] and Zhang et al.'s [42].

Table 2. The description of the traditional machine learning models and the parameter settings.

Model	Description
Naive Bayes (NB)	- Probabilistic classifier based on the Bayes' theorem - Assumes the independence between features
Decision Tree (DT)	- C4.5 classifier using the J48 algorithm - Confidence factor for pruning: 0.25 - Minimum number of instances: 1 - The number of decimal places: 1 - Determines the amount of data used for pruning: 5 - Reduced error pruning: True
Support Vector Machine (SVM)	- Non-probabilistic binary classification model that finds a decision boundary with a maximum distance between two classes - Kernel: RBF - Exponent = 1.0, Complexity (c) = 10.0
Random Forest (RF)	- Kind of ensemble model that generates final result by incorporating the results of multiple decision trees - Maximum tree depth: 30 - Number of iterations: 150 - Each tree has no depth limitation

In order to optimize the CNN model proposed in this paper, we experimented with various settings, and the optimal settings were as follows: (1) the embedding dimension $E = 25$; (2) C_1 and C_2 were set to 3; (3) K_1 and K_2 were 16 and 8; and (4) the pooling layer dimension $P = 2$. We set the strides for the kernels to be 1. Every layer took the Rectified Linear Unit (ReLU) as an activation function except for the output layer that took a softmax function. The number of the trainable parameter was 476,331, and we adopted Adam's optimizer with an initial learning rate of 0.001. We took the L2 regularization for the fully-connected layer. The number of epochs was changed according to the complexity of the network (i.e., the number of layers and parameters). For example, the first network was trained for 5 epochs, and the model of Kim [1] was trained for 10 epochs.

In the study of Zhang et al. [42], NB-SVM was the best. The effectiveness of the combination with different models (e.g., NB, SVM, NB-SVM) may vary for different tasks. We tested different combinations with our datasets and discovered that the SVM with the RBF kernel was the best enhanced model for our task. Thus, the CNN enhanced by SVM was used for comparison with other models.

5. Result and Discussion

5.1. Result

The three datasets (movie review data, customer review data, and Stanford Sentiment Treebank data) were applied to the proposed CNN models, traditional machine-learning models, and other state-of-the-art models. Note that, we conducted two experiments: binary classification and ternary classification. Tables 3 and 4 show the experimental results of binary classification, where the two values for each cell correspond to the "positive" and "negative" classes, respectively. Table 5 shows the experimental results of ternary classification in MR data. Table 3 shows the experimental results with

the MR data, while Table 4 describes the experimental results with the CR and SST data. These tables include the accuracy, precision, recall, F1 score, and weighted-F1 score. For example, the F1 scores of the decision tree are 67.2% for positive and 31.1% for negative in Table 3.

Table 3. The results with the MR data, where Emb, Conv, Pool, globalpool, FC stand for embedding layer, convolutional layer, pooling layer, global pooling layer, and fully-connected layer, respectively.

Model	Accuracy	Precision	Recall	F1	Weighted-F1
Decision Tree	59.64	58.0/64.0	76.2/39.6	67.4/47.1	57.2
Naive Bayes	56.40	57.5/53.3	77.7/30.7	66.1/38.9	52.0
Support Vector Machine	54.95	57.7/55.2	93.0/9.1	69.3/15.5	44.9
Random Forest	58.73	56.4/59.8	74.7/39.4	66.4/46.4	58.1
Kim [1]	80.85	80.7/80.9	76.2/84.7	78.3/82.8	80.75
Zhang et al. [42]	77.28	72.4/69.1	56.1/82.1	63.2/75.0	69.62
Emb+Conv+Conv+Pool+FC	81.06	81.5/80.7	75.6/85.6	78.4/83.1	80.96
Emb+Conv+Pool+FC	79.70	77.4/81.63	78.2/80.9	77.8/81.3	79.71
Emb+Conv+Conv+Conv+Pool+FC	80.30	80.2/80.4	75.3/84.5	77.7/82.4	80.26
Emb+Conv+Pool+Conv+FC	78.17	74.4/81.8	79.5/77.1	76.8/79.4	78.22
Emb+Conv+globalpool+FC	77.54	77.3/77.7	71.8/82.4	74.4/79.9	77.39
Emb+Conv+Conv+globalpool+FC	79.06	79.1/79.0	73.5/83.8	76.2/81.3	78.98
Emb+Conv+Pool+Conv+Pool+FC	79.11	78.6/79.5	74.3/83.1	76.4/81.2	79.0
Emb+Conv+Pool+Conv+Pool+Conv+Pool+FC	74.61	84.1/72.8	59.5/90.6	69.7/80.7	75.7

Table 4. The results with the CR and SST data, where Emb, Conv, Pool, globalpool, FC stand for embedding layer, convolutional layer, pooling layer, global pooling layer, and fully-connected layer, respectively.

Model	CR		SST	
	Weighted-F1	F1	Weighted-F1	F1
Decision Tree	63.7	47.0/74.5	51.5	62.4/41.7
Naive Bayes	61.0	77.7/31.8	35.7	10.6/63.4
Support Vector Machine	59.7	26.5/78.5	37.8	68.6/4.0
Random Forest	64.4	41.8/76.9	51.2	59.0/47.3
Kim [1]	74.8	78.6/65.6	56.1	47.2/66.3
Zhang et al. [42]	54.8	64.7/37.7	52.1	45.6/59.5
Emb+Conv+Conv+Pool+FC	78.3	84.8/67.1	68.3	70.5/65.7
Emb+Conv+Pool+FC	78.3	82.3/71.3	66.5	67.7/65.1
Emb+Conv+Conv+Conv+Pool+FC	70.4	82.0/50.3	68.2	68.4/68.0
Emb+Conv+Pool+Conv+FC	75.3	84.0/60.3	68.6	70.5/66.5
Emb+Conv+globalpool+FC	81.4	86.1/73.4	70.2	72.8/67.2
Emb+Conv+Conv+globalpool+FC	79.4	84.5/70.5	70.0	71.2/68.7
Emb+Conv+Pool+Conv+Pool+FC	73.18	82.8/56.5	66.62	67.7/65.4
Emb+Conv+Pool+Conv+Pool+Conv+Pool+FC	51.57	77.6/6.7	65.21	70.2/59.5

In Table 3, the Random Forest (RF) achieved the best F1 scores, while the Decision Tree (DT) exhibited comparable results. Our first network was the best among all models including the state-of-the-art models of [1,42]. Similarly, in Table 4, the RF and the DT achieved the best F1 scores among the traditional models, but our fifth network outperformed all other models. Note that the best networks in Tables 3 and 4 were different. That is, the first network was the best with the MR data, which had relatively longer sentences, as shown in Table 3, whereas the fifth network was the best for the other two datasets (e.g., CR, SST) that had relatively shorter sentences.

One may argue that stacking more convolutional layers might be better, as the deeper network is known to capture higher level patterns. This might be true, but it should be noted that the deeper network is not always better than the shallow networks. The depth of the network needs to be determined according to the data characteristics; too deep networks will probably overfit, whereas too

shallow networks will underfit. For example, although the third network was deeper than the first network, the first network outperformed the third network, as shown in Table 3. To clarify this, we conducted experiments varying the number of convolutional layers from 1–5, and its result is depicted in Figure 2. Meanwhile, the first network can be compared with the fourth network to show that stacking two consecutive convolutional layers is better. The fifth and sixth networks may be compared with the first and second networks, if one can see if the max-pooling and the global max-pooling contribute to the models. The seventh and eighth networks had similar structures to that in [44,45].

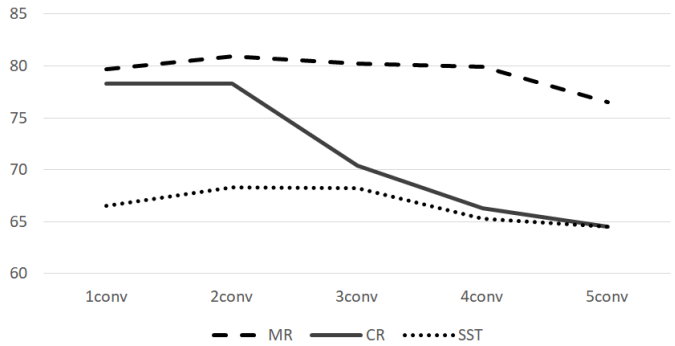

Figure 2. Classification performance according to the depth of the convolutional layer, where the horizontal axis means the number of stacked convolutional layers.

We also conducted ternary classification. The binary classification assumed that there were only two labels (e.g., positive and negative), whereas the ternary classification assumed that there were three labels (e.g., positive, negative, and neutral). As shown in Table 5, our first network was the best for the ternary classification, but the performance fell to a 68.31% weighted-F1 score.

Table 5. Performance of ternary classification with the MR data, where Emb, Conv, Pool, globalpool, FC stand for embedding layer, convolutional layer, pooling layer, global pooling layer, and fully-connected layer, respectively.

Model	Weighted-F1
Decision Tree	46.8
Naive Bayes	27.2
Support Vector Machine	34.9
Random Forest	46.2
Kim [1]	68.2
Zhang et al. [42]	67.5
Emb+Conv+Conv+Pool+FC	68.3
Emb+Conv+Pool+FC	55.0
Emb+Conv+Conv+Conv+Pool+FC	64.6
Emb+Conv+Pool+Conv+FC	66.9
Emb+Conv+globalpool+FC	65.3
Emb+Conv+Conv+globalpool+FC	67.5

5.2. Discussion

5.2.1. Comparison with Other Models

In terms of the F1 scores, our network was about 10% greater than the traditional machine models. We believe that the biggest reason is the inherent ability of the CNN to capture local patterns. The CNN has the ability to capture local patterns and higher level patterns through its convolutional and pooling layers, as explained in [48]. We believe that such an ability of the CNN model produces the performance gap. The proposed network was also better than the state-of-the-art deep learning models

such as those of Kim [1] and Zhang et al. [42]. Kim [1] and Zhang et al. [42] commonly utilized three filters (sizes of 3, 4, and 5) in parallel, in the hope to capture the filters' arbitrary features with different sizes. On the other hand, our network had consecutive filters that had two benefits: (1) hierarchical (higher) features were extracted, and (2) the high-level filters would have a broader range to see local patterns, where such advantages of consecutive filters were already reported in many previous studies [56–58]. The three-way approach, employed in [42], gave worse performance than the model of [1], although it was designed to combine of a CNN model and a traditional machine learning model. The reason can be explained by the fact that the CNN model itself was already sufficient to achieve the high performance, as shown in Table 3, so the combination with other traditional models (e.g., Naive Bayes) degraded the overall performance. In Zhang et al. [42], among the sentences classified by CNN [1], the sentences with weak confidence were re-classified by the traditional machine learning models, and it improved the overall performance. On the other hand, for the three datasets in this paper, we found that the ratio of weakly-classified instances was biased, as shown in Figure 3; this means that the entropy of the weakly-classified instances was small, so it was difficult to expect that the performance would improve by the combination with the traditional models.

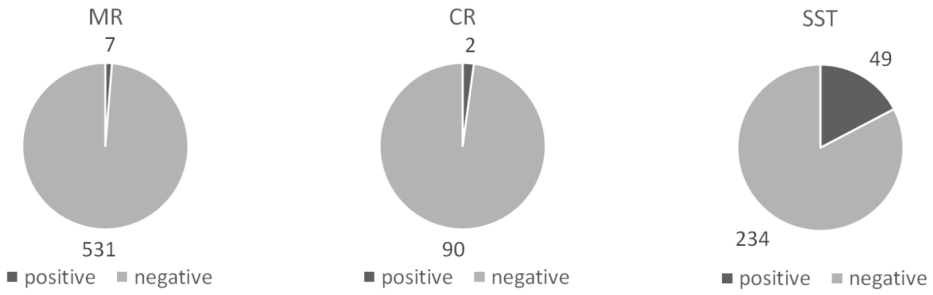

Figure 3. The ratio of positive/negative sentences with weak confidence.

5.2.2. Network Structure

As shown in Tables 3 and 4, the max-pooling had better performance in MR data, whereas the global max-pooling seemed helpful in the CR data and the SST data. The max-pooling layer gave the largest value in a certain subarea as an output, while the global max-pooling did this in the whole area. Figure 4 shows the difference.

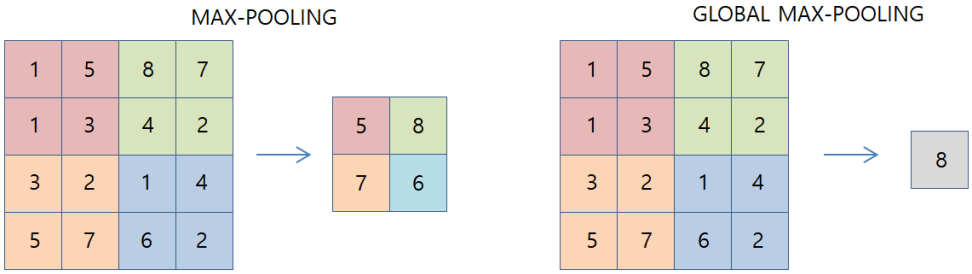

Figure 4. The difference of max-pooling and global max-pooling.

If MR data adopted global max-pooling, the large number of missing values would be discovered. Because the length of the total sentence of the MR data was longer than that of CR data and SST data, the performance of the MR data was lower. On the other hand, CR data and SST data with relatively a short length of sentences were not degraded even when using global max-pooling and, in some cases, performed better than using max-pooling. As a result, CR data and SST data showed the highest performance when one convolutional layer and global max-pooling layer were combined. The reason is also that they were shorter and simpler data than the MR data, so they showed higher performance in relatively monotonous models. However, The network with two convolutional layers never lagged

behind the traditional machine learning models and state-of-the-art models in terms of performance in this experimental result.

6. Conclusions

In this paper, we designed a convolutional neural network for the sentiment classification. By experimental results, we showed that the consecutive convolutional layers contributed to better performance on relatively long text. The proposed CNN models achieved about 81% and 68% accuracies for the binary classification and ternary classification, respectively. As future work, we will apply our work to other classification tasks (e.g., gender classification). We will also continue finding better structures for the sentiment classification; for example, residual connection for stacking more layers.

Author Contributions: Conceptualization, H.K.; validation, H.K., Y.J.; investigation, H.K.; resources, Y.J.; data curation, H.K.; writing–original draft preparation, H.K.; writing–review and editing, Y.J.; visualization, H.K.; supervision, Y.J.; project administration, Y.J.

Acknowledgments: This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIP; Ministry of Science, ICT & Future Planning) (No. 2019021348). This work was supported by the Soonchunhyang University Research Fund.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
2. Nal, K.; Grefenstette, E.; Blunsom, P. A convolutional neural network for modelling sentences. *arXiv* **2014**, arXiv:1404.2188.
3. Lei, T.; Barzilay, R.; Jaakkola, T. Molding cnns for text: Non-linear, non-consecutive convolutions. *arXiv* **2015**, arXiv:1508.04112.
4. Amazon Movie Review Dataset. Available online: https://www.kaggle.com/ranjan6806/corpus2#corpus/ (accessed on 11 November 2012).
5. Movie Review Dataset. Available online: https://www.kaggle.com/ayanmaity/movie-review#train.tsv/ (accessed on 11 November 2012).
6. Rotten Tomatoes Movie Review Dataset. Available online: https://www.kaggle.com/c/movie-review-sentiment-analysis-kernels-only/ (accessed on 11 November 2012).
7. Consumer Reviews Of Amazon Products Dataset. Available online: https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products/ (accessed on 11 November 2012).
8. Stanford Sentiment Treebank Dataset. Available online: https://nlp.stanford.edu/sentiment/code.html/ (accessed on 11 November 2012).
9. Pak, A.; Paroubek, P. Twitter as a corpus for sentiment analysis and opinion mining. *LREc* **2010**, *10*, 1320–1326.
10. Alm, C.O.; Roth, D.; Sproat, R. Emotions from text: Machine learning for text-based emotion prediction. In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, Vancouver, BC, Canada, 6–8 October 2005; pp. 579–586.
11. Bartlett, M.S.; Littlewort, G.; Frank, M.; Lainscsek, C.; Fasel, I.; Movellan, J. Recognizing facial expression: Machine learning and application to spontaneous behavior. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 2, pp. 568–573.
12. Heraz, A.; Razaki, R.; Frasson, C. Using machine learning to predict learner emotional state from brainwaves. In Proceedings of the Seventh IEEE International Conference on Advanced Learning Technologies (ICALT 2007), Niigata, Japan, 18–20 July 2007; pp. 853–857.
13. Yujiao, L.; Fleyeh, H. Twitter Sentiment Analysis of New IKEA Stores Using Machine Learning. In Proceedings of the International Conference on Computer and Applications, Beirut, Lebanon, 25–26 July 2018.

14. Read, J. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In Proceedings of the ACL Student Research Workshop, Ann Arbor, Michigan, 27–27 June 2005; pp. 43–48.
15. Kennedy, A.; Inkpen, D. Sentiment classification of movie reviews using contextual valence shifters. *Comput. Intell.* **2006**, *22*, 110–125.
16. Jiang, L.; Yu, M.; Zhou, M.; Liu, X.; Zhao, T. Target-dependent twitter sentiment classification. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, Oregon, 19–24 June 2011; Volume 1, pp. 151–160.
17. Neethu, M.; Rajasree, R. Sentiment analysis in twitter using machine learning techniques. In Proceedings of the 2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT), Tiruchengode, India, 4–6 July 2013; pp. 1–5.
18. Gautam, G.; Yadav, D. Sentiment analysis of twitter data using machine learning approaches and semantic analysis. In Proceedings of the 2014 Seventh International Conference on Contemporary Computing (IC3), Noida, India, 7–9 August 2014; pp. 437–442.
19. Tripathy, A.; Agrawal, A.; Rath, S.K. Classification of sentiment reviews using n-gram machine learning approach. *Expert Syst. Appl.* **2016**, *57*, 117–126.
20. Wan, X. A comparative study of cross-lingual sentiment classification. In Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology, Macau, China, 4–7 December 2012; Volume 1, pp. 24–31.
21. Akaichi, J. Social networks' Facebook'statutes updates mining for sentiment classification. In Proceedings of the 2013 International Conference on Social Computing, Alexandria, VA, USA, 8–14 September 2013; pp. 886–891.
22. Denecke, K. Using sentiwordnet for multilingual sentiment analysis. In Proceedings of the 2008 IEEE 24th International Conference on Data Engineering Workshop, Cancun, Mexico, 7–12 April 2008; pp. 507–512.
23. Bahrainian, S.A.; Dengel, A. Sentiment analysis using sentiment features. In Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Atlanta, GA, USA, 17–20 November 2013; Volume 3; pp. 26–29.
24. Antai, R. Sentiment classification using summaries: A comparative investigation of lexical and statistical approaches. In Proceedings of the 2014 6th Computer Science and Electronic Engineering Conference (CEEC), Colchester, UK, 25–26 September 2014; pp. 154–159.
25. Hasan, A.; Moin, S.; Karim, A.; Shamshirband, S. Machine learning-based sentiment analysis for twitter accounts. *Math. Comput. Appl.* **2018**, *23*, 11.
26. Mensikova, A.; Mattmann, C.A. Ensemble sentiment analysis to identify human trafficking in web data. Workshop on Graph Techniques for Adversarial Activity Analytics (GTA 2018), Marina Del Rey, CA, USA, February 2018
27. Pang, B.; Lee, L. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Barcelona, Spain, 21–26 July 2004; pp. 271–278.
28. Valakunde, N.; Patwardhan, M. Multi-aspect and multi-class based document sentiment analysis of educational data catering accreditation process. In Proceedings of the 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, Pune, India, 15–16 November 2013; pp. 188–192.
29. Yassine, M.; Hajj, H. A framework for emotion mining from text in online social networks. In Proceedings of the 2010 IEEE International Conference on Data Mining Workshops, Sydney, Australia, 13 December 2010; pp. 1136–1142.
30. Karamibekr, M.; Ghorbani, A.A. A structure for opinion in social domains. In Proceedings of the 2013 International Conference on Social Computing, Alexandria, VA, USA, 8–14 September 2013; pp. 264–271.
31. Ghiassi, M.; Lee, S. A domain transferable lexicon set for Twitter sentiment analysis using a supervised machine learning approach. *Expert Syst. Appl.* **2018**, *106*, 197–216.
32. Balazs, J.A.; Velásquez, J.D. Opinion mining and information fusion: A survey. *Inf. Fusion* **2016**, *27*, 95–110.
33. Chaturvedi, I.; Cambria, E.; Welsch, R.E.; Herrera, F. Distinguishing between facts and opinions for sentiment analysis: Survey and challenges. *Inf. Fusion* **2018**, *44*, 65–77.
34. Dong, L.; Wei, F.; Tan, C.; Tang, D.; Zhou, M.; Xu, K. Adaptive recursive neural network for target-dependent twitter sentiment classification. In Proceedings of the 52nd Annual Meeting of the Association for

Computational Linguistics (Volume 2: Short Papers), Baltimore, Maryland, USA, 23–25 June 2014; Volume 2, pp. 49–54.
35. Huang, M.; Cao, Y.; Dong, C. Modeling rich contexts for sentiment classification with lstm. *arXiv* **2016**, arXiv:1605.01478.
36. Tang, D.; Qin, B.; Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1422–1432.
37. Qian, Q.; Huang, M.; Lei, J.; Zhu, X. Linguistically regularized lstms for sentiment classification. *arXiv* **2016**, arXiv:1611.03949.
38. Mikolov, T. Statistical language models based on neural networks. PhD thesis, Brno University of Technology, 2012.
39. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv* **2014**, arXiv:1412.3555.
40. Tai, K.S.; Socher, R.; Manning, C.D. Improved semantic representations from tree-structured long short-term memory networks. *arXiv* **2015**, arXiv:1503.00075.
41. Wang, F.; Zhang, Z.; Lan, M. Ecnu at semeval-2016 task 7: An enhanced supervised learning method for lexicon sentiment intensity ranking. In Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016), San Diego, California, 16–17 June 2016; pp. 491–496.
42. Zhang, Y.; Zhang, Z.; Miao, D.; Wang, J. Three-way enhanced convolutional neural networks for sentence-level sentiment classification. *Inf. Sci.* **2019**, *477*, 55–64.
43. Severyn, A.; Moschitti, A. Unitn: Training deep convolutional neural network for twitter sentiment classification. In Proceedings of the 9th International Workshop On Semantic Evaluation (SemEval 2015), Denver, Colorado, 4–5 June 2015; pp. 464–469.
44. Deriu, J.; Lucchi, A.; De Luca, V.; Severyn, A.; Müller, S.; Cieliebak, M.; Hofmann, T.; Jaggi, M. Leveraging large amounts of weakly supervised data for multi-language sentiment classification. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017; pp. 1045–1052.
45. Ouyang, X.; Zhou, P.; Li, C.H.; Liu, L. Sentiment analysis using convolutional neural network. In Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, UK, 26–28 October 2015; pp. 2359–2364.
46. Socher, R.; Huval, B.; Manning, C.D.; Ng, A.Y. Semantic compositionality through recursive matrix-vector spaces. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island, Korea, 12–14 July 2012; pp. 1201–1211.
47. Rios, A.; Kavuluru, R. Convolutional neural networks for biomedical text classification: Application in indexing biomedical articles. In Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics, Atlanta, Georgia, 9–12 September 2015; pp. 258–267.
48. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28*; Neural Information Processing Systems Foundation, Inc., Montreal, Quebec, Canada, 2015; pp. 649–657.
49. Yih, W.t.; He, X.; Meek, C. Semantic Parsing for Single-Relation Question Answering. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Baltimore, Maryland, USA, 23–25 June 2014; Volume 2, pp. 643–648.
50. Shen, Y.; Xiaodong, H.; Gao, J.; Deng, L.; Mensnil, G. Learning Semantic Representations Using Convolutional Neural Networks for Web Search. In Proceedings of the 23rd International Conference on World Wide Web, Seoul, Korea, 7–11 April 2014; pp. 373–374.
51. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, Austin, Texas, 25–30 January 2015.
52. Gehring, J.; Auli, M.; Grangier, D.; Dauphin, Y.N. A convolutional encoder model for neural machine translation. *arXiv* **2016**, arXiv:1611.02344.
53. Dos Santos, C.; Gatti, M. Deep convolutional neural networks for sentiment analysis of short texts. In Proceedings of the COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, Dublin, Ireland, 23–29 August 2014; pp. 69–78.

54. Boser, B.E.; Guyon, I.M.; Vapnik, V.N. A training algorithm for optimal margin classifiers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, Pennsylvania, USA, 27–29 July 1992; pp. 144–152.
55. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32.
56. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision And Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
57. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2818–2826.
58. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, California, USA, 4–9 February 2017.