Neural Computing and Applications (2021) 33:17425–17438
https://doi.org/10.1007/s00521-021-06328-5

ORIGINAL ARTICLE

An attention-based CNN-LSTM model for subjectivity detection in opinion-mining

Santwana Sagnika[1] · Bhabani Shankar Prasad Mishra[1] · Saroj K. Meher[2]

Received: 7 February 2021 / Accepted: 10 July 2021 / Published online: 22 July 2021

Abstract
Opinion-mining generally refers to analyzing opinions on various topics available in the form of text. It is an essential operation of natural language processing since it enables efficient decision-making and planning for users and businesses. Opinion-mining can be made more comfortable and more effective by initially performing subjectivity detection, i.e., identifying the text as subjective or objective. An opinion-mining model can better identify the opinions present in the remaining subjective statements by removing objective statements. With this reasoning, we present an efficient subjectivity detection model for improved accuracy in Opinion-mining. The model uses a strategic combination of convolutional neural network (CNN) and long short-term memory (LSTM). CNN and LSTM are state-of-the-art deep learning models that can efficiently process textual data and identify inherent connections and patterns with varying abstraction levels. The proposed work combines the strengths of both these models in an ensemble model. Effectiveness of the model is enhanced with the incorporation of an attention network. In the present task, the sentences are represented as word embeddings that include sentiment information and part-of-speech. The proposed model is applied on two movie review datasets, and its performance is evaluated compared with state-of-the-art methods. Various performance indexes have validated the superiority of the proposed model in the opinion-mining task.

Keywords Subjectivity detection · Sentiment analysis · Natural language processing · Opinion-mining · Deep learning · Convolutional neural network · Long short-term memory network · Attention network

1 Introduction

The amount of user-generated data is rapidly increasing on the Internet. It is due to the popularity of social media and e-commerce, which gives users readily available platforms to share their views and opinions. Users can post their opinions regarding various products, services, events, issues, etc. These opinions can be accessed publicly from e-commerce platforms like Amazon and Flipkart and review aggregators like IMDb and Tripadvisor. They are especially significant since users worldwide refer to them before making decisions regarding purchasing products (e.g., electronics, books, clothing) or providing services (e.g., restaurants, movies, repair services).

On the other hand, the product sellers and service providers use these opinions and reviews as feedback and improve their services. The reviews are also utilized in strategic planning for future products and services. Hence, it becomes essential to analyze these opinions for improving business policies. Thus, a massive amount of such reviews demands an automatic process of analysis for quick and informative pattern discovery. *Opinion-mining* or *sentiment analysis* is one such process that provides valuable information to the users. Several research attempts have been made in this domain that involve

✉ Santwana Sagnika
santwana.sagnikafcs@kiit.ac.in

Bhabani Shankar Prasad Mishra
bsmishrafcs@kiit.ac.in

Saroj K. Meher
saroj.meher@isibang.ac.in

[1] School of Computer Engineering, Kalinga Institute of Industrial Technology Deemed To Be University, Bhubaneswar, Odisha 751024, India

[2] Systems Science and Informatics Unit, Indian Statistical Institute, Bangalore Centre, 8th Mile, Mysore Road, RVCE Post, Bangalore, Karnataka 560059, India

multiple subtasks, such as review usefulness analysis, named entity recognition, word disambiguation, negation handling, spam detection, and subjectivity detection [1, 2, 37].

Subjectivity detection refers to the identification of text as fact or opinion. Most reviews or articles consist of a mix of factual and opinionated content. Factual statements are neutral, i.e., they do not express any opinion of the user. Hence, they do not contribute to the task of Opinion-mining. Only the opinionated statements convey the user's judgment regarding the product/service in question, therefore determining the Opinion-mining process results [3]. For example, in a mobile phone review, the statement "The phone has a 6 GB RAM and a 5.5inch HD display." is a factual statement.

In contrast, the statement "Good product, genuine battery life, fast processor." is opinionated and more useful to readers. Thus, subjectivity detection helps filter out the factual content from a piece of text. The remaining text, which is entirely opinionated content, can then be subjected to further processing. By performing subjectivity detection as a preliminary step, the amount of content to be analyzed can be reduced without affecting the results. Further, the presence of unbiased reviews and texts can dilute the detection task. Therefore, identifying and filtering them through subjectivity detection can improve the final detection [4, 36]. Hence, it makes sense to approach subjectivity detection as a significant research problem and explore various models to find a solution.

To perform subjectivity detection, this work applies an attention-based CNN-LSTM classification model. State-of-the-art deep learning models like Convolutional Neural Networks (CNN) and Long Short-Term Memory Networks (LSTM) are being extensively applied to solve many problems involving various kinds of data, like image, text, video, speech, etc., to produce accurate results. CNN models explore the spatial connections of the data, while LSTM models retain the temporal associations. Combinatorial models (CNN-LSTM) discover both aspects and have been proven to generate better performance. By incorporating an attention mechanism into a combinatorial model, specific parts of the data are focused on, which relieves the model from the burden of focusing on the real data. It renders the model highly effective and fast [5, 6]. The proposed model is applied on a movie reviews dataset and evaluated compared to various other efficient models.

Further, the model is used on a different movie reviews dataset to perform subjectivity detection and filter the objective sentences out. Sentiment analysis is then performed on the shortened reviews. The results are compared with the sentiment analysis results on the original dataset to demonstrate the usefulness of subjectivity detection towards sentiment analysis.

The contributions of this work are—

- An attention-based CNN-LSTM model is proposed, and applied to a text detection task.
- An improved Word2Vec word embedding mechanism is used for feature representation, and its suitability for subjectivity detection is demonstrated.
- The benefits of using subjectivity detection as a preliminary step of sentiment analysis are assessed.
- The proposed CNN-LSTM-based subjectivity detection in the sentiment analysis task is applied on various data sets and verified with different performance measurement indices.

The rest of the paper is organized as follows. In Sect. 2, the existing literature in subjectivity detection and the application of attention-based deep learning models is discussed. Details of the basic concepts and architecture of CNN, LSTM, and attention networks are described in Sect. 3. Section 4 elaborates on the proposed work, including the data representation and the proposed model's architecture. In Sect. 5, we discuss the datasets and experimental setup. Section 6 presents the proposed model's results and describes subjectivity detection in the sentiment analysis task. Section 7 concludes and suggests the future scope of the proposed work.

2 Literature review

The area of subjectivity detection has been relatively less explored than most other text mining tasks. We discuss some notable work in the field. Wiebe and Riloff [12] created rule-based subjective and objective classifiers, which were trained only on un-annotated data and learned extraction patterns, rivaling existing classifiers in performance over articles from the world press. Xuan et al. [8] explored the syntactic structures of text and discovered specific syntax-based patterns involving adjectives, adverbs, verbs, and noun and verb extensions. These patterns can extract informative linguistic features, which, combined with traditional features, gave impressive results on a movie review dataset. Keshavarz and Abadeh [7] created a subjectivity lexicon MHSubLex using a genetic algorithm-based technique. They used it to perform subjectivity detection on three Twitter datasets by extracting subjective and objective statements. Research has also been done on multilingual [9] and cross-lingual [10] subjectivity detection.

Machine learning and deep learning techniques have gained rapid popularity in the past decade, owing to their efficiency, adaptability, and ability to extract useful patterns without human intervention. Subjectivity detection has highly benefited from the application of various

machine learning and deep learning techniques. Lin et al. [13] performed subjectivity detection at a sentence level, using a hierarchical Bayesian model, which used latent Dirichlet Allocation (subjLDA). It was a weakly supervised approach relying on a limited set of linguistic clues. It gave comparable results against techniques that used larger training sets on the multi-perspective question answering (MPQA) corpus. Kamal [11] applied supervised machine learning using four different classifiers by considering linguistic characteristics such as term frequency, opinion seed word, negation, etc., for unigram characterization. These characteristics were further used to mine for feature-opinion pairs from a product review dataset.

Wang and Manning [35] used fast dropout training on Naïve Bayes-based classifiers to experiment on various datasets and obtained 93.6% accuracy on subjectivity detection for movie reviews. Kim [34] tested on CNN networks for various sentence-based classification tasks and used pre-trained word embeddings for the purpose. They achieved an accuracy of 93% for subjectivity detection using multichannel CNN. Chaturvedi et al. [14] used CNN pre-trained on network patterns of words and concepts extracted from the text using dynamic Gaussian Bayesian networks. They achieved 5–10% improvements over accuracies of baseline techniques on the MPQA and movie review datasets. Chaturvedi et al. [15] further applied an extreme learning framework that incorporated Bayesian networks to build interconnections. They used a fuzzy recurrent network to model temporal features. Their experiments on multiple datasets provided useful output and proved the model's ability to port to other domains and languages. Rustamov [16] also worked on sentence-level subjectivity classification, proposing a hybrid model of the Fuzzy Control system, the Hidden Markov model, and Adaptive neuro-fuzzy inference system. He employed a pruned inverse document frequency (IDF) weighting function for feature detection, achieving higher accuracy than any individual classifier model.

The addition of attention mechanism has dramatically enhanced the performance of deep models like CNN and LSTM. Some of the significant efforts are discussed. Zhao and Wu [17] used an attention-based CNN for sentence classification that modeled long-term word correlation and contextual information on the TREC sentence dataset and achieved competitive results with standard CNN models. Yang et al. [18] applied an attention mechanism to bidirectional LSTM in two models for target-dependent sentiment classification of a Twitter dataset and achieved improvement over baseline techniques. Zhang et al. [19] also approached target-dependent sentiment classification using a multi-layer CNN with an attention mechanism that modeled context representation and achieved high accuracy on product reviews and Twitter data. Gan et al. [20] designed a dilated CNN based on sparse attention to perform targeted sentiment analysis, which identified sentiment orientation and handled complex sequences. Their experiments on Twitter and reviews datasets gave a noticeable improvement in performance over various standard and hybrid LSTM and CNN models. Attention-based LSTM enhanced by sentiment information was utilized by Lei et al. [21] to perform sentiment analysis on review datasets. They obtained improved accuracy over other LSTM variants.

Zhang et al. [22] combined a recurrent neural network (RNN) and CNN with an attention layer. They utilized it to carry out relation classification, achieving a comparable F1 score with other LSTM models. For Chinese sentiment classification, Liu et al. [23] proposed a novel architecture by hybridizing attention mechanism, CNN, and gated recurrent unit (GRU), which effectively obtained context and semantic information and improved the overall performance. Another subtask of sentiment analysis, sarcasm detection, was also addressed using a softmax attention layer added to a bidirectional LSTM and CNN combination. Jain et al. [24] used this model on a dataset of bilingual tweets to learn semantic contexts and perform highly accurate real-time sarcasm detection.

A detailed study on the afore-mentioned techniques led us to identify the research gap in subjectivity detection. Even though it is a vital text processing task and a good step in sentiment analysis, little work exists. The recent popularity of deep learning models in text processing, and the enhancement in their capabilities due to attention mechanism, motivated us to apply such a model to perform efficient subjectivity detection. Our model identifies and utilizes context and semantic information over a long-range. To the best of our knowledge, the proposed model is a novel approach to subjectivity detection.

3 Basic concepts

In this section, we discuss the concepts required to understand the work proposed in this paper. We discuss the standard architectures of CNN and LSTM and the idea of attention mechanism, and its contribution to a model's effectiveness.

3.1 Convolutional neural networks (CNN)

CNN's are multilayered feed-forward neural architectures. Initially developed for image processing applications, CNN and its variants have evolved into a quick and efficient tool for natural language processing tasks. A CNN performs a series of convolution and pooling operations, followed by a fully connected neural network. These models can detect

the presence of patterns in the input, irrespective of their position. Over the repeated layers, these patterns can be identified at higher levels, leading to the detection of meaningful data features.

CNN uses a convolution layer, which acts as a filter and runs over the input, applying a mathematical function. In the case of text data, the filter is generally 1-dimensional, applied to the sentence's length. This process picks up the significant parts of the input and generates a feature map. The pooling layer computes representative values from the feature map. It does so by considering the most considerable value of activation in the given area. This process performs data reduction, as well as finds abstract higher-level features from the input. Typically there are multiple pairs of sequential convolutional and pooling layers. The last pooling layer is linked to a fully connected layer that performs classification on the obtained representational data, just like a classical neural network model [29, 31]. Figure 1 presents the general architecture of a CNN model.

3.2 Long short-term memory networks (LSTM)

LSTMs are a variant of Recurrent Neural Networks (RNN). RNNs are powerful and dynamic systems that are suitable for sequential inputs, like text and speech. They can accurately predict the next word or character in a text. These models process the input text unit by unit and maintain states at each hidden layer. This is made possible due to the looped structure of the network, in which the neural network's output is connected as an input to itself. Such chain-like arrangement enables them to retain a history of the inputs already processed. Figure 2 presents the general architecture of an RNN model. At a given time step t, the input to the network s is i_t, and the output is o_t. The self-loop can be unrolled to see the chain-like connection.

LSTMs use special hidden units to retain long-term memory. There are four neural components in each repeating unit. They are the forget gate layer, the input gate layer (consisting of an update component and an addition component), and the output gate layer. These are connected to the cell state, i.e., the special memory cell which accumulates and passes information throughout the model. The connection is made through gates, which control the addition and modification of data to the cell state. The forget gate layer decides the amount of information to be removed. The input gate layer combines the values to be updated and the candidates to be added. The output gate layer determines the parts of the internal memory state that are to be provided as output [29, 30]. The internal structure of an LSTM network is presented in Fig. 3.

Consider a LSTM unit at a given time step t. Here, in_t represents input gate, f_t is the forget gate, op_t is the output gate. The internal cell state is represented by c_t, and o_t is the output of the unit, known as the hidden state.

The processing within the LSTM is controlled by the following equations.

$$in_t = \sigma\left(W_{ip} i_t + U_{ip} o_{t-1} + b_{ip}\right) \quad (1)$$

$$f_t = \sigma\left(W_f i_t + U_f o_{t-1} + b_f\right) \quad (2)$$

$$op_t = \sigma\left(W_{op} i_t + U_{op} o_{t-1} + b_{op}\right) \quad (3)$$

$$\tilde{c}_t = \tanh\left(W_c i_t + U_c o_{t-1} + b_c\right) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + in_t \odot \tilde{c}_t \quad (5)$$

$$o_t = op_t \odot \tanh(c_t) \quad (6)$$

σ represents the logistic sigmoid function and $\odot$ performs element to element multiplication. W and U are corresponding weight vectors, while b represents the corresponding bias vectors. $\tilde{c}_t$ is the vector of candidate values for the memory cell state, from which the cell state c_t is obtained by combining with the input gate [38].

Equation (1) passes the previous output o_{t-1} and current input i_t through a sigmoid layer to combine the relevant information to be added. It represents the input gate. The forget gate, represented in Eq. (2), uses a similar mechanism to forget specific parts of the information with some amount of probability. Equation (3) represents the output

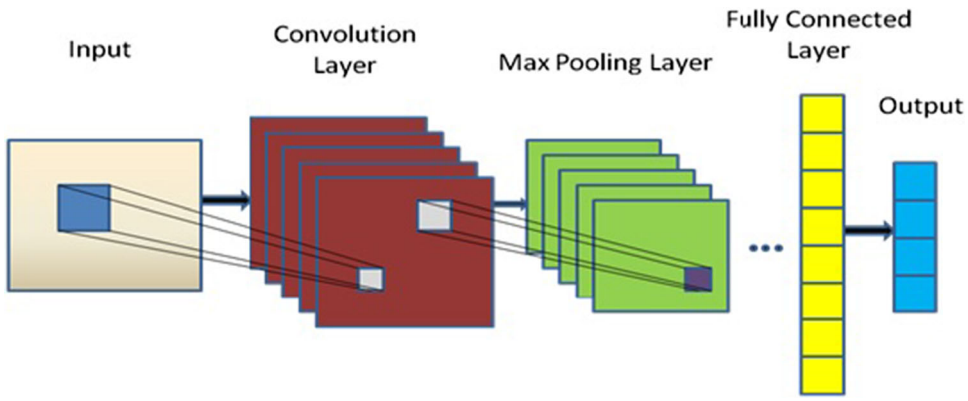

Fig. 1 Architecture of Convolutional Neural Network

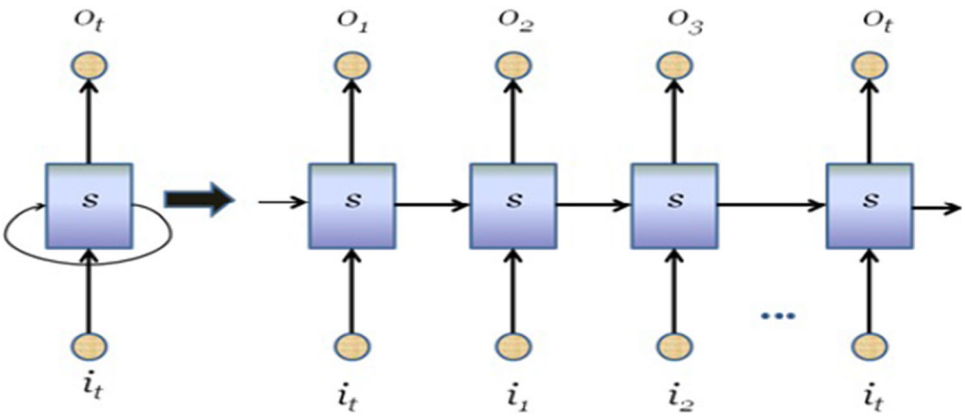

Fig. 2 Architecture of Recurrent Neural Network

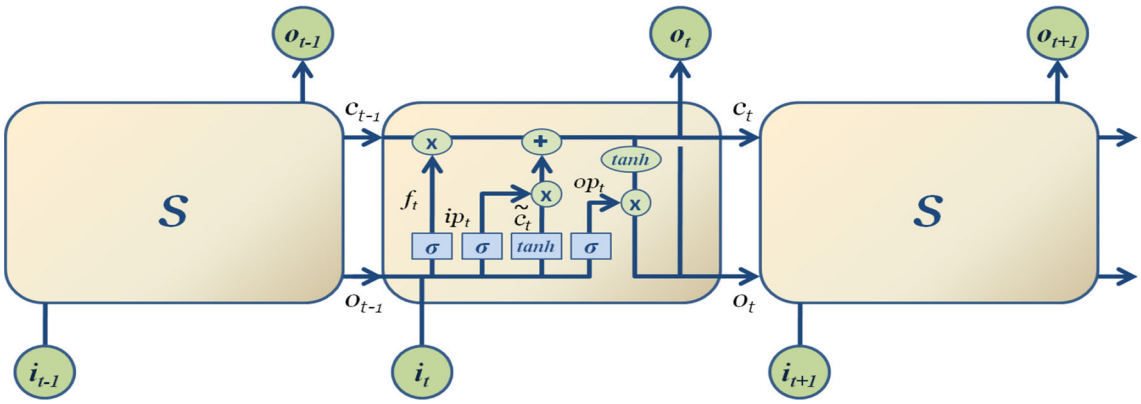

Fig. 3 Architecture of Long Short-Term Memory Network

gate, which again implements a similar method as the previous two gates. It uses a sigmoid layer to decide the states to be forwarded. Equation (4) is the current moment information ($\tilde{c}_t$), which is obtained by applying a *tanh* function on the previous output and current input. This is the new information that is to be added. All these gates use their respective weight vectors *(W and U)*, and bias vectors *(b)*.

In Eq. (5), the current memory information and input gate information, i.e., output of Eqs. (1) and (4), are multiplied. This is added to the multiplication of the forget gate information with the long-term memory information of the previous step, i.e., c_{t-1}. This generates the long-term memory information of the current step (c_t). Finally, this information passes through a *tanh* layer and is multiplied by the result of the output gate in Eq. (3), to generate the output of the LSTM (o_t).

3.3 Attention networks

The attention mechanism has expanded and improved the application of deep learning models in various fields, especially in natural language processing. Attention mechanism addresses very long-range dependency problems in LSTMs. With the increase in the length of input sentences, the performance of an LSTM decreases due to its reduced ability to remember connections between words that are too far apart within a sentence [25]. Besides, LSTMs cannot prioritize individual sections of the sentence that are relevant. Attention mechanism can solve both these problems. It creates context vectors by considering all input words and attaches relative weightage to them [26].

The first attention model was proposed by Bahdanau et al. [27]. Figure 4 illustrates the architecture presented in work. The model uses a bidirectional RNN as an encoder. The encoder produces a set of states, which are annotations representing the words in the input sentence. Every state contains information about the entire sentence while focusing more on a particular phrase and its surrounding parts. It helps keep track of the words throughout the sentence, i.e., focus attention on them.

The encoder states are combined with the current decoder state (initially, the last encoder state) to perform training on a feed-forward neural network. The neural network learns and generates differential scores for the encoder states to represent attention. Then, attention weights are generated by applying the softmax function on

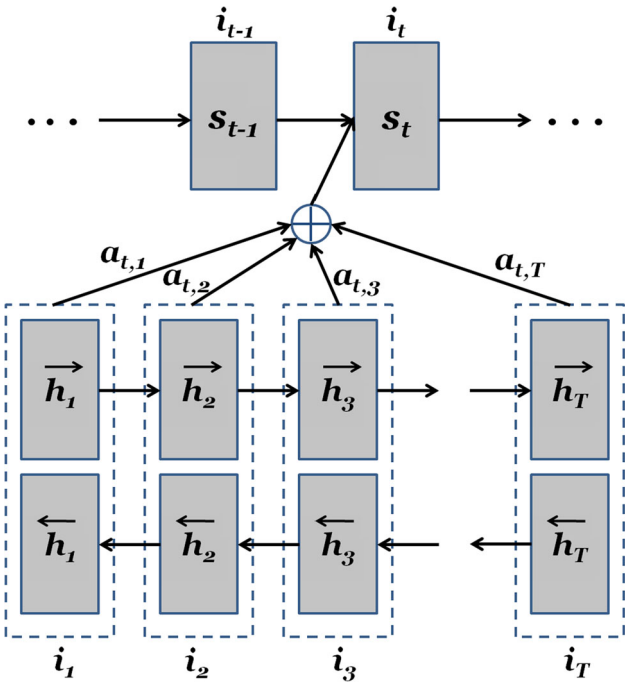

Fig. 4 Architecture of Attention Network

the encoder state scores. Using these attention weights, a vector representing the input sentence is calculated, known as context vector. Thus, the attention network can generate a context vector using the states, unlike RNN, which generates a fixed context vector using only the last form.

The decoder, which is a gated recurrent unit (GRU), uses the generated context vector and the output of the prior time step (i.e., the operation performed at the last instant of time) to predict the next word of the sentence, which also becomes the current state of the decoder. This current state is once again combined with the encoder states and used to retrain the feed-forward neural network for computing the encoder state scores in the next time step. This process continues till the decoder generates an "END" token, i.e., the entire sentence is generated [28].

Let $x = (x_1,x_2,\ldots,x_{Tx})$ represent an input sequence, where x_j represents a word. The annotation of x_j is given by

$$h_j = \overrightarrow{h_j^T} ; \overleftarrow{h_j^T} \tag{7}$$

which is the concatenation of a forward hidden state $\overrightarrow{h_j}$ and a backward hidden state $\overleftarrow{h_j}$.

The bidirectional RNN generates a set of left-to-right forward hidden states $\left(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{Tx}}\right)$ and a set of right-to-left backward hidden states $\left(\overleftarrow{h_1}, \overleftarrow{h_2}, \ldots, \overleftarrow{h_{Tx}}\right)$.

At time step i, the context vector is generated by the weighted sum of the annotations using the equation

$$c_i = \sum_{j=1}^{Tx} \alpha_{ij} h_j \tag{8}$$

The weight α_{ij} of each annotation h_j is calculated through a softmax function by

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{Tx} \exp(e_{ik})} \tag{9}$$

α_{ij} is a probability calculation. It signifies the importance of annotation h_j in deciding the next hidden state s_i from the previous hidden state s_{i-1}. Thus c_i is the expected annotation, based on all annotations with probabilities α_{ij}.

e_{ij} is the output of an alignment model $a()$ that maps the input at step j and the output at step i. It is a score generated by matching the jth annotation of the input (i.e., h_j) with the output of the hidden state s_{i-1}.

$$e_{ij} = a\left(s_{i-1}, h_j\right) \tag{10}$$

The alignment model is a feed-forward neural network that calculates a soft alignment, allowing the backpropagation of the gradient of the cost function. The probability calculation is the implementation of attention mechanism in the decoder. The information is spread out among the annotation sequence, to be retrieved selectively by the decoder.

Though initially attention mechanism was proposed as an efficient solution to machine translation, it has hence been effectively used for a variety of text mining tasks. The performance of LSTM-based models has improved significantly with the inclusion of attention mechanism. This has led us to incorporate attention mechanism in our proposed model for this work.

4 Proposed work

In this paper, the proposed work uses a model that combines CNN and LSTM with attention mechanism. The model takes sentences as input and classifies them as subjective or objective. Since this model performs classification at the sentence-level, any text which is to be analyzed (e.g., review, tweet, document, etc.) is broken down into a set of sentences, which are individually classified. The aggregate classification of the sentences in a body of text can give its overall subjectivity classification. In this section, we explain the input preprocessing and representation steps, and the details of the proposed architecture.

4.1 Data representation

The input data are divided into a set of sentences. Every sentence is represented using a fixed size numeric vector.

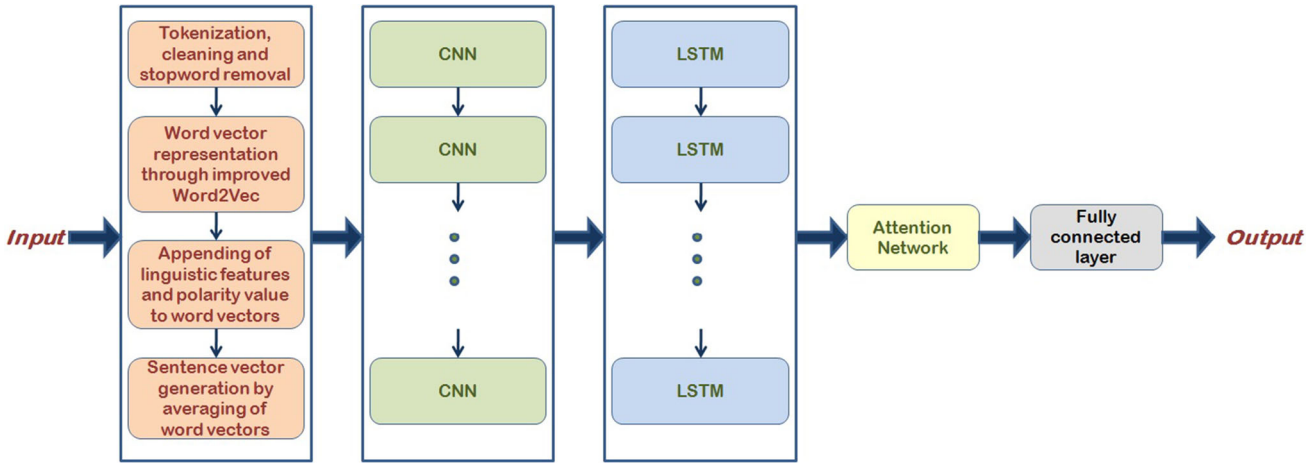

Fig. 5 Proposed system architecture

This vector representation is known as word embedding. Word embeddings for all words can be generated from the input data by various techniques. Two neural-network-based word embeddings techniques have gained popularity recently, namely Word2Vec and GloVe. These are models that train on the input data and generate vectors of the same size for all the text words. Such pre-trained word embeddings are also available online, which are trained on large corpora of generalized data. These pre-trained embeddings can be used as generalized embeddings for any dataset instead of generating new word embeddings. In our previous work [32], we have modified the generalized word embeddings using sentiment-related information to generate improved word embeddings. These improved word embeddings are verified to be more suitable for sentiment analysis than generalized word embeddings. Since subjectivity detection is closely related to sentiment analysis, we have used the improved word embedding method to generate word embeddings for this work.

The technique uses generalized Word2Vec and GloVe embeddings for affect words (words that represent emotions) and performs clustering on them using self-organizing maps. Based on these clusters, the words are iteratively moved towards sentimentally similar words, and farther from sentimentally dissimilar words. This movement is performed by particle swarm optimization technique, which uses the inter-word distance to control the actions. In this work, we have used 100-dimensional improved Word2Vec word embeddings generated from the dataset.

For any sentence, first, the stopwords are removed. Then the word embedding is generated for each word. The generated word embedding of 100 dimensions is embedded with additional information about linguistic features and polarity. The linguistic features consist of 5 values, each representing a part-of-speech, i.e., adjective, state verb, direct action verb, interpretive action verb, and a state action verb. The part of speech that is most suitable for the current word is set to 1, and the remaining categories are set to 0. If none of them match, all values are set to 0. One more number is added, which represents the polarity value. The polarity of the word is obtained from an existing polarity corpus and appended. After appending values for linguistic features and polarity, each phrase's length embedding increases to 106 dimensions. The generated embeddings of all words in a sentence are averaged. This averaged vector is taken as an input vector to the proposed model. The model is described subsequently.

4.2 Model architecture

Our proposed model comprises a CNN-LSTM design with added attention method. The architecture of the proposed model is shown in Fig. 5.

The first block represents the data preprocessing steps as described before. Once the data are preprocessed, it is fed to the model for training. The model begins with the second block in the figure, which represents the CNN block. We have taken consecutive pairs of convolutional and max pooling layers. These layers are connected to a set of consecutive LSTM networks, as shown in the third block in the figure. The capability of these LSTM layers are enhanced by an attention network, which performs the attention mechanism described before on the LSTM output. The attention network's output is fed to a fully connected layer that generates the final output.

The functionalities of each block are discussed in the previous section. The CNN-LSTM combination is a well-established model that acts as an encoder-decoder arrangement. The CNN does the encoding by feature extraction and dimension reduction. Multiple layers of CNN provide different levels of abstractions in the form of low-level features (word level), mid-level features (phrase

level), and high-level features (sentence level). The LSTM uses these extracted features to perform the detection. Thus, the model could extract the spatial information, i.e., the one-dimensional structure of words in a sentence, and the temporal information, i.e., the order of words in texts. The attention network enables long-range information retention, and the fully connected layer executes the task of classification.

In the studied model, we have used a total of nine layers. The first layer is a 1D convolution layer with 128 filters, having a kernel size of ten and a relu (rectified linear) activation function. A 1D max-pooling layer of pool size six is applied. This step is followed by another 1D convolution layer with 64 filters having a kernel size of five and a 1D max-pooling layer of pool size four. Then, there is the last 1D convolution layer with 32 filters, having kernel size of three. There is no need for any more pooling layers because the output dimension is reduced to a single vector of length 32. This is fed to an LSTM layer of 128 units with a dropout of 30% and recurrent dropout of 20%. The next layer is another LSTM with 64 units with similar dropout rates. An attention layer is applied to this, which generates an output vector of size 64. The final layer is a dense layer, which takes the input vector of 64 and uses the sigmoid activation function to generate the final classification output. The model is compiled with an adam optimizer using accuracy as the metric. Figure 6 shows a snippet of the Python code used to build the model.

5 Implementation

5.1 Dataset

We have used the Subjectivity dataset by Pang and Lee [33] to train the model. This dataset consists of 10,000 sentences obtained from movie reviews available on Rotten Tomatoes, and plot summaries from the Internet Movie Database (IMDb) website. The sentences from the reviews are labeled as subjective, and the sentences from the plots are labeled as objective. The dataset is equally distributed, i.e., there are 5000 subjective and 5000 objective sentences. All sentences are in lowercase and consist of at least ten words.

5.2 Experimental setup

The experiments are performed using Python 3.5 on an Intel i5 desktop with 32 GB RAM and 2.71 GHz frequency. The significant packages are NLTK for NLP tasks, Keras, Theano, and Tensorflow for implementation of deep architectures, and Scikit-learn for machine learning models and performance measurement indices.

For performing classification, the entire dataset is randomly divided into two sets. One set is used to train the model, and the other is used to test the efficiency. Three different train-test ratios (80%:20%, 70%:30%, and 60%:40%) are prepared for conducting the experiment. Each classification model is trained and tested with the same train-test splits for ten rounds. The average of the 10 readings is presented in the tables in the results section. Such random splitting tests the model performance's consistency over different sets of training and testing data over multiple iterations. The model's performance is compared against standard machine learning and deep learning models, and some existing techniques obtained from the literature study.

5.3 Performance measurement indices

For presenting the results, different performance measurement indices are used in this work, namely Precision, Recall, Accuracy, Kappa Score. These parameters rely on the parameters True Positives, False Positives, True negatives, and False Negatives. They are explained as follows.

True Positives is the number of items predicted as true, which are correct predictions. *False Positives* is the number of items predicted as true, which are wrong predictions. *True Negatives* is the number of items predicted as false, which are correct predictions. *False Negatives* is the number of items predicted as false, which are wrong predictions. *Precision* is the ratio of positive items correctly classified to the total number of positive data items in the dataset. *Recall* is the ratio of positive items correctly classified to the total number of data items provided in the dataset. *Accuracy* denotes the ratio of correct predictions to the total number of data items provided in the dataset. It represents the overall ability of the model. *Kappa score* is a comparison of the accuracy of the model with the accuracy of a random system. It measures the match between the classified items and the items labeled as ground truth, thus controlling items that might be correctly classified by chance. A kappa value of 1 represents a perfect match, whereas a kappa value of 0 represents no match. The formula for kappa score is given by Eq. 11.

$$\text{KappaScore} = \frac{N(\text{TruePositives} + \text{True Negatives}) - X}{N^2 - X} \tag{11}$$

where,

$$\begin{aligned} X = {} & (\text{True Positives} + \text{False Positives}) \\ & * (\text{TruePositives} + \text{False Negatives}) \\ & + (\text{True Negatives} + \text{False Negatives}) \\ & * (\text{True Negatives} + \text{False Positives}) \end{aligned} \tag{12}$$

```python
class attention(Layer):
    def __init__(self,**kwargs):
        super(attention,self).__init__(**kwargs)

    def build(self,input_shape):
        self.W=self.add_weight(name="att_weight",shape=(input_shape[-1],1),initializer="normal")
        self.b=self.add_weight(name="att_bias",shape=(input_shape[1],1),initializer="zeros")
        super(attention, self).build(input_shape)

    def call(self,x):
        et=K.squeeze(K.tanh(K.dot(x,self.W)+self.b),axis=-1)
        at=K.softmax(et)
        at=K.expand_dims(at,axis=-1)
        output=x*at
        return K.sum(output,axis=1)

    def compute_output_shape(self,input_shape):
        return (input_shape[0],input_shape[-1])

    def get_config(self):
        return super(attention,self).get_config()
```

(a)

```python
#Attention LSTM
inputs=Input(shape=(feature_vector.shape[1],1))
l1=Conv1D(128, 10, input_shape=(106,1),activation = 'relu')(inputs)
l11=MaxPooling1D(pool_size = 6)(l1)
l2=Conv1D(64, 5, activation = 'relu')(l11)
l21=MaxPooling1D(pool_size = 4)(l2)
l3=Conv1D(32, 3, activation = 'relu')(l21)
l4=LSTM(128,return_sequences=True,dropout=0.3,recurrent_dropout=0.2)(l3)
att_in = LSTM(64,return_sequences=True,dropout=0.3,recurrent_dropout=0.2)(l4)
att_out=attention()(att_in)
outputs=Dense(1,activation='sigmoid',trainable=True)(att_out)
classifier=Model(inputs,outputs)
classifier.compile(loss='binary_crossentropy', optimizer='adam', metrics=['acc'])
classifier.summary()
```

(b)

Fig. 6 Snapshots of Python code for **a** attention mechanism **b** building the model

Here, N denotes the total number of data items in the dataset.

In addition, we have used the *standard deviation of accuracy* (A_{std}) and the *standard deviation of Kappa score* (K_{std}) over the ten rounds of each model in order to analyze the statistical consistency in performance. This analysis demonstrates the variation of the model performances, and a lower value of standard deviation signifies higher stability. We have also used Receiver Operating Characteristic (ROC) curves to provide graphical analysis. ROC curve plots true positive rate and false positive rate. The area under the curve (AUC) represents an aggregate of the performance, i.e., more the area, better the model.

6 Results and discussion

6.1 Subjectivity analysis

Table 1 shows the experimental results that are obtained by applying the proposed model on the subjectivity dataset. The results of various other classifiers are incorporated for comparative purposes. It includes results of three different train-test split ratios. The corresponding precision, recall,

Table 1 Performance comparison of the proposed attention-based CNN-LSTM model and existing models on Subjectivity dataset (Class 0 represents objective and Class 1 represents subjective sentences)

Model	Precision		Recall		Accuracy	Kappa score	A_{std}	K_{std}
	Class 0	Class 1	Class 0	Class 1				
Train-test ratio – 80:20								
Random forest	0.84	0.81	0.81	0.84	0.828	0.682	0.0135	0.0132
CNN	0.87	0.94	0.95	0.86	0.907	0.817	0.0160	0.0148
LSTM	0.87	0.90	0.91	0.86	0.879	0.763	0.0212	0.0155
CNN-LSTM	0.91	0.96	0.97	0.90	0.938	0.877	0.0171	0.0139
Attention-based CNN-LSTM (Proposed)	0.96	0.98	0.98	0.96	0.971	0.942	0.0090	0.0130
Train-test ratio – 70:30								
Random forest	0.77	0.86	0.89	0.73	0.807	0.617	0.0142	0.0219
CNN	0.86	0.92	0.93	0.84	0.886	0.777	0.0157	0.0210
LSTM	0.88	0.86	0.86	0.88	0.874	0.751	0.0200	0.0207
CNN-LSTM	0.90	0.93	0.94	0.90	0.921	0.839	0.0177	0.0157
Attention-based CNN-LSTM (proposed)	0.94	0.97	0.97	0.93	0.958	0.920	0.0098	0.0151
Train-test ratio – 60:40								
Random Forest	0.82	0.80	0.78	0.84	0.803	0.609	0.0153	0.0246
CNN	0.86	0.87	0.88	0.86	0.868	0.743	0.0162	0.0245
LSTM	0.83	0.87	0.88	0.81	0.848	0.697	0.0191	0.0235
CNN-LSTM	0.87	0.91	0.92	0.84	0.886	0.756	0.0166	0.0228
Attention-based CNN-LSTM (proposed)	0.90	0.93	0.93	0.89	0.911	0.825	0.0092	0.0174

accuracy and kappa score values of each experiment are provided. The standard deviations of the accuracy and kappa score are given as A_{std} and K_{std}, respectively.

For performance comparison of the proposed model (attention-based CNN-LSTM), we have used classification models, such as random forest, CNN, LSTM, and hybrid CNN-LSTM. The random forest classifier is an ensemble classifier comprising of multiple decision trees. It is selected for comparative analysis because of its robustness and simplicity of use. The other models have been discussed in the previous sections.

The results demonstrate increased values in each of the performance indices. The split ratio of 80:20 gives the best results, where the proposed model reaches an accuracy of 0.971 and a Kappa score of 0.942. Split ratios of 70:30 and 60:40 give accuracies of 0.958 and 0.911, and Kappa scores of 0.920 and 0.825, respectively. This performance is higher than the performance of other models that we have tested, which include standard machine learning and deep learning models, such as random forest, CNN, LSTM, and a CNN-LSTM combinatorial model. The respective precision and recall values are also higher in the proposed model.

The standard deviations in both accuracy and kappa score are lower in the proposed model, as compared to the existing models taken for comparison. This shows that the proposed model is also consistent, i.e., it gives similar results when tested over different combinations of training and testing sets for a fixed ratio. The values also show minimal change for different train-test split ratios, which proves that the model is robust and does not change its consistency when the size of training set is varied.

The incorporation of attention mechanism in our proposed model improves its performance, which is because it remembers long range relationships and dependencies within the sentences. In the context of subjectivity analysis, the connection between various parts of the sentences is retained in a better way, which helps to relate objects and opinions present far from each other. This leads to an improvement in the prediction results. The combined power of CNN, LSTM and attention mechanism prove advantageous in sentence-based subjectivity analysis.

In Fig. 7, we have compared the accuracy obtained by our proposed model to the best state-of-the-art models found from the literature survey, which have been applied on the movie reviews dataset we have used. The Bayesian deep belief CNN (BCDBN) proposed in [14] showed the highest accuracy of 96.4% yet, followed by a multi-channel CNN [34] with an accuracy of 93.6% and the Naïve Bayes support vector machine (NB-SVM) model used in [35] having 93.2% accuracy. With a top accuracy of 0.971, i.e.,

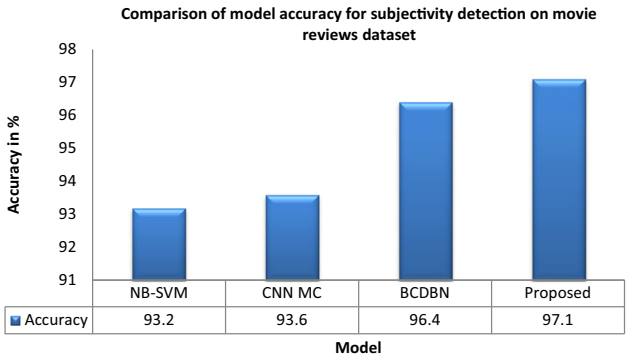

Fig. 7 Comparison of accuracy of proposed model with existing models for subjectivity detection on the movie reviews dataset

97.1%, our proposed model outperforms these three existing models.

Figure 8 shows the box plots of Kappa scores for the three train-test split ratios. Figure 9 shows the box plots of all the classifiers for different train-test split ratios. Kappa score explains how much better the classifier performs relative to a classifier that randomly guesses, based on the frequency of each class. A higher value of Kappa score indicates stronger confidence in the classifier model. The proposed model gives a range of Kappa scores above 0.75, which is considered as substantial. A comparative view shows that the proposed model gives a higher range of Kappa score in all three split ratios. As the training set size increases, the box plot shows shifting of Kappa score towards 1, which denotes increasing confidence in the model. For 60:40 split, the median is at 0.825. For 70:30 split, the median falls at 0.92, and for 80:20 split, the median lies at 0.94. These values highlight the reliability of the proposed model and also show that the model performance gets better with larger training sets.

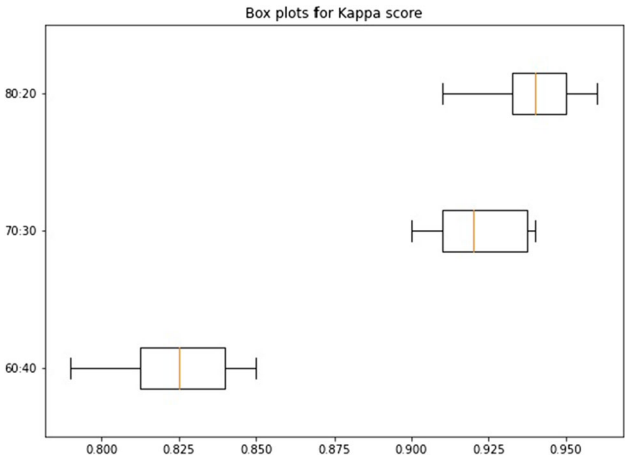

Fig. 8 Box plots of Kappa score obtained by the proposed model for train-test split ratios of 80:20, 70:30 and 60:40

6.2 Sentiment analysis

Based on the identification of the subjective and objective sentences, a body of text can be filtered to only contain subjective sentences. This can help to improve the accuracy of performing sentiment analysis on it. To verify this premise, we have used the model trained on the subjectivity corpus in the previous step. The trained model is applied on an IMDb dataset consisting of 25,000 movie reviews. The reviews are split into sentences and each sentence is classified as subjective or objective using the proposed model. The objective sentences are removed and the reviews are reconstructed using only the subjective sentences. These modified reviews are then classified using various classifiers, and compared with the classification results of the original reviews. The preprocessing and data preparation is performed in the same way as the subjectivity classification in the previous step. Table 2 gives the results obtained.

The results in Table 2 compare the performance metrics of sentiment analysis directly on the IMDb dataset, with sentiment analysis after performing subjectivity detection on each review and filtering out the objective statements. It is evident that performing subjectivity detection as a prior step to sentiment analysis leads to an improvement in the performance. By removing objective sentences, the text irrelevant to sentiment analysis is reduced. As a result, the reviews retain only significant subjective information that contains the opinion tone. This is better understood by the classifier. Thus, higher accuracy is achieved. There is also a marginal increase in the training time of the classifier since the length of many reviews is reduced. Our proposed attention-based CNN-LSTM model also gives the best performance in sentiment analysis, as compared to standard models like random forest, CNN, and LSTM classifiers. The standard deviation of accuracy of the proposed model is comparatively lower, which shows that the model is consistent and gives minimal variations over different contents of the dataset.

Figure 10 presents the receiver-operating characteristics (ROC) curves that compare the performance of sentiment analysis after filtering of objective sentences with sentiment analysis performed without any filtering. The curves of 80:20, 70:30 and 60:40 train-test splits are plotted separately. The area under the curve (AUC) for each of these cases is higher for classification after filtering than classification on the original dataset. For 80:20 split, an increase of 0.04 is obtained. Similarly, an increase of 0.03 is observed for 70:30 split, and the 60:40 split shows and increase of 0.03. This proves that there is a significant increase in the number of correct classifications after the objective sentences are identified and removed from the

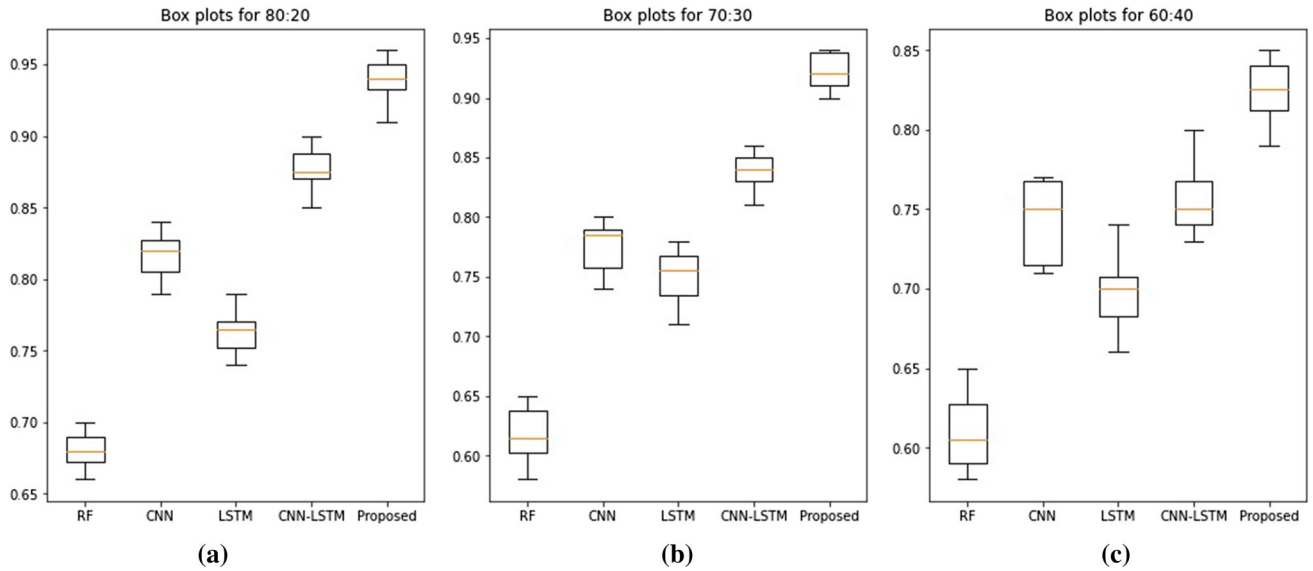

Fig. 9 Box plots of Kappa scores obtained by different classifiers for train-test split ratios of **a** 80:20, **b** 70:30 and **c** 60:40

Table 2 Performance comparison of sentiment analysis on original dataset and dataset with objective sentences filtered

Model	Classification on original dataset					Classification on dataset with objective sentences filtered				
	Precision	Recall	Accuracy	Kappa score	A_{std}	Precision	Recall	Accuracy	Kappa score	A_{std}
Train-test ratio – 80:20										
Random forest[a]	0.83	0.83	0.832	0.665	0.0040	0.84	0.84	0.838	0.682	0.0031
CNN	0.85	0.85	0.854	0.709	0.0077	0.88	0.88	0.880	0.756	0.0068
LSTM	0.84	0.84	0.838	0.677	0.0021	0.87	0.87	0.871	0.743	0.0019
CNN-LSTM	0.87	0.87	0.866	0.736	0.0071	0.91	0.90	0.910	0.822	0.0060
Attention-based CNN-LSTM (Proposed)	0.91	0.91	0.917	0.825	0.0030	0.95	0.95	0.947	0.893	0.0022
Train-Test Ratio – 70:30										
Random Forest	0.84	0.84	0.841	0.676	0.0051	0.87	0.87	0.863	0.740	0.0038
CNN	0.86	0.86	0.858	0.728	0.0063	0.89	0.89	0.889	0.784	0.0050
LSTM	0.84	0.84	0.842	0.740	0.0026	0.88	0.87	0.878	0.767	0.0021
CNN-LSTM	0.88	0.88	0.880	0.762	0.0080	0.91	0.91	0.910	0.819	0.0068
Attention-based CNN-LSTM (proposed)	0.89	0.89	0.885	0.776	0.0038	0.93	0.93	0.933	0.855	0.0024
Train-test ratio – 60:40										
Random forest	0.85	0.84	0.848	0.688	0.0052	0.86	0.86	0.858	0.720	0.0048
CNN	0.86	0.86	0.861	0.734	0.0063	0.89	0.88	0.885	0.795	0.0055
LSTM	0.83	0.83	0.832	0.727	0.0021	0.85	0.85	0.848	0.701	0.0018
CNN-LSTM	0.87	0.87	0.870	0.743	0.0068	0.90	0.89	0.895	0.778	0.0061
Attention-based CNN-LSTM (proposed)	0.88	0.88	0.879	0.757	0.0041	0.91	0.91	0.901	0.790	0.0030

original reviews. Thus, our work shows that subjectivity detection plays an important role in enhancing the performance of sentiment analysis on reviews.

7 Conclusion and future work

Attention-based networks are the state-of-the-art versions of LSTM networks that have significantly improved their effectiveness and analysis powers. In this work, we have

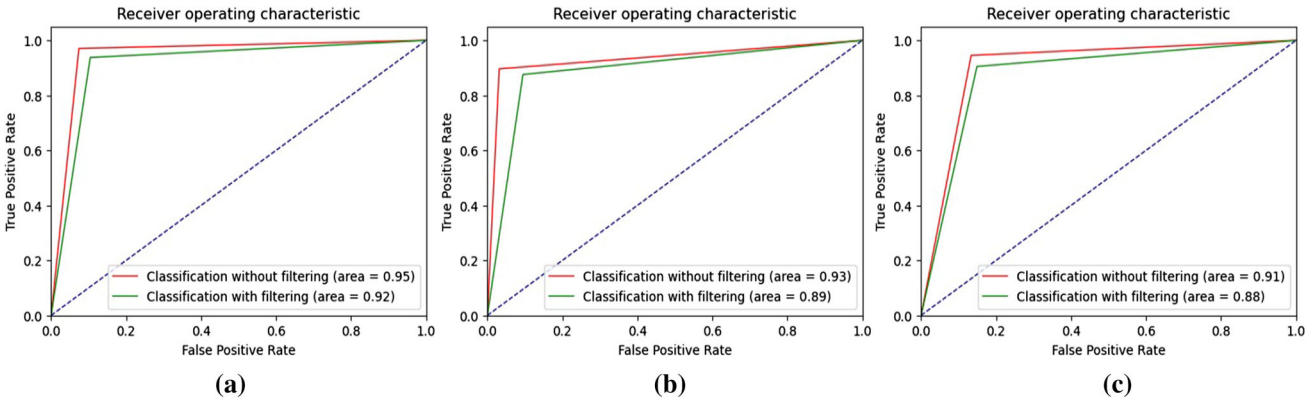

Fig. 10 Comparison of ROC curves for sentiment analysis with objective filtering and sentiment analysis without filtering for train-test split ratio of **a** 80:20 **b** 70:30 and **c** 60:40

tested their efficacy on subjectivity detection, a vital task in the field of Opinion-mining. The combined power of CNN-LSTM networks and attention mechanism in the proposed model has shown improved accuracy, precision, recall, and kappa score. As a further step, subjectivity detection's contribution as an initial step of sentiment analysis is also verified by filtering the objective sentences. Sentiment analysis on the filtered reviews is more accurate than original reviews. The proposed model also gives better results than other classification models in sentiment analysis. This shows the suitability of the model to Opinion-mining tasks in general. Future work can be done in further fine-tuning the model and increasing its accuracy and training time. The model can also be applied to other natural language processing tasks, such as review usefulness analysis, spam detection, and cross-domain Opinion-mining since these problems are similarly approachable.

Funding This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

1. Ravi K, Ravi V (2015) A survey on opinion-mining and sentiment analysis: tasks, approaches and applications. Knowl-Based Syst 89:14–46
2. Sagnika S, Pattanaik A, Mishra BSP, Meher SK (2020) A review on multi-lingual sentiment analysis by machine learning methods. J Eng Sci Technol Rev 13(2):154–166
3. Chaturvedi I, Cambria E, Welsch RE, Herrera F (2018) Distinguishing between facts and opinions for sentiment analysis: Survey and challenges. Inf Fusion 44:65–77
4. Satapathy R, Chaturvedi I, Cambria E, Ho SS, Na JC (2017) Subjectivity detection in nuclear energy tweets. Comput Sist 21(4):657–664
5. Bhunia AK, Konwer A, Bhunia AK, Bhowmick A, Roy PP, Pal U (2019) Script identification in natural scene image and video frames using an attention based Convolutional-LSTM network. Pattern Recogn 85:172–184
6. Fang Y, Gao J, Huang C, Peng H, Wu R (2019) Self multi-head attention-based convolutional neural networks for fake news detection. PLoS ONE 14(9):e0222713
7. Keshavarz HR, Saniee Abadeh M (2018) MHSubLex: Using metaheuristic methods for subjectivity classification of micro-blogs. J AI Data Min 6(2):341–353
8. Xuan HNT, Le AC (2012) Linguistic features for subjectivity classification. In: 2012 international conference on asian language processing, pp 17–20. IEEE
9. Banea C, Mihalcea R, Wiebe J (2014) Sense-level subjectivity in a multilingual setting. Comput Speech Lang 28(1):7–19
10. Amini I, Karimi S, Shakery A (2019) Cross-lingual subjectivity detection for resource lean languages. In: Proceedings of the tenth workshop on computational approaches to subjectivity, sentiment and social media analysis, pp 81–90
11. Kamal A (2013) Subjectivity classification using machine learning techniques for mining feature-opinion pairs from web opinion sources. arXiv preprint arXiv:1312.6962
12. Wiebe J, Riloff E (2005) Creating subjective and objective sentence classifiers from unannotated texts. International conference on intelligent text processing and computational linguistics. Springer, Berlin, pp 486–497
13. Lin C, He Y, Everson R (2011) Sentence subjectivity detection with weakly-supervised learning. In: Proceedings of 5th international joint conference on natural language processing, pp 1153–1161
14. Chaturvedi I, Cambria E, Poria S, Bajpai R (2016) Bayesian deep convolution belief networks for subjectivity detection. In: 2016 IEEE 16th international conference on data mining workshops (ICDMW), pp 916–923. IEEE
15. Chaturvedi I, Ragusa E, Gastaldo P, Zunino R, Cambria E (2018) Bayesian network based extreme learning machine for subjectivity detection. J Franklin Inst 355(4):1780–1797
16. Rustamov, S. (2018). A hybrid system for subjectivity analysis. *Advances in fuzzy systems*, *2018*.
17. Zhao Z, Wu Y (2016) Attention-based convolutional neural networks for sentence classification. In: INTERSPEECH, pp 705–709

18. Yang M, Tu W, Wang J, Xu F, Chen X (2017) Attention based LSTM for target dependent sentiment classification. In: Thirty-First AAAI conference on artificial intelligence
19. Zhang S, Xu X, Pang Y, Han J (2020) Multi-layer attention based CNN for target-dependent sentiment classification. Neural Process Lett 51(3):2089–2103
20. Gan C, Wang L, Zhang Z, Wang Z (2020) Sparse attention based separable dilated convolutional neural network for targeted sentiment analysis. Knowledge-Based Syst 188:104827
21. Lei Z, Yang Y, Yang M (2018) Sentiment lexicon enhanced attention-based LSTM for sentiment classification. In: Thirty-second AAAI conference on artificial intelligence
22. Zhang X, Chen F, Huang R (2018) A combination of RNN and CNN for attention-based relation classification. Proc Comput Sci 131:911–917
23. Liu J, Yang Y, Lv S, Wang J, Chen H (2019) Attention-based BiGRU-CNN for Chinese question classification. *J Ambient Intell Hum Comput*, pp 1–12
24. Jain D, Kumar A, Garg G (2020) Sarcasm detection in mash-up language using soft-attention based bi-directional LSTM and feature-rich CNN. *Appl Soft Comput*, p 106198
25. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y (2014) Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078
26. Attention Mechanism in Deep Learning | Attention Model Keras. Available: https://www.analyticsvidhya.com/blog/2019/11/comprehensive-guide-attention-mechanism-deep-learning/
27. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473
28. Intuitive Understanding of Attention Mechanism in Deep Learning | by Harshall Lamba | Towards Data Science. Available: https://towardsdatascience.com/intuitive-understanding-of-attention-mechanism-in-deep-learning-6c9482aecf4f
29. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
30. Liu P, Qiu X, Huang X (2016) Recurrent neural network for text classification with multi-task learning. arXiv preprint arXiv:1605.05101
31. Arel I, Rose DC, Karnowski TP (2010) Deep machine learning-a new frontier in artificial intelligence research [research frontier]. IEEE Comput Intell Mag 5(4):13–18
32. Sagnika S, Mishra BSP, Meher SK (2020) Improved method of word embedding for efficient analysis of human sentiments. Multim Tools Appl 79:32389–32413
33. Pang B, Lee L (2004) A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. arXiv preprint cs/0409058
34. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882
35. Wang S, Manning C (2013) Fast dropout training. In: International conference on machine learning, pp 118–126
36. Das N, Sagnika S (2020) A subjectivity detection-based approach to sentiment analysis. Machine learning and information processing. Springer, Singapore, pp 149–160
37. Sindhu C, Sasmal B, Gupta R, Prathipa J (2021) Subjectivity detection for sentiment analysis on twitter data. Artificial intelligence techniques for advanced computing applications. Springer, Singapore, pp 467–476
38. Islam MZ, Islam MM, Asraf A (2020) A combined deep CNN-LSTM network for the detection of novel coronavirus (COVID-19) using X-ray images. Inf Med Unlocked 20:100412