Journal of Computer and Communications, 2022, 10, 112-131
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Towards Mining Public Opinion: An Attention-Based Long Short Term Memory Network Using Transfer Learning

G. M. Sakhawat Hossain[1], Md. Harun Or Rashid[2], Md. Rafiqul Islam[2], Ananya Sarker[2], Must. Asma Yasmin[2]

[1]Department of Computer Science and Engineering, Rangamati Science and Technology University, Rangamati, Bangladesh
[2]Department of Computer Science and Engineering, Bangladesh Army University of Engineering and Technology, Qadirabad, Bangladesh
Email: u19mcse005p@student.cuet.ac.bd, gmsakhawathossain@gmail.com

How to cite this paper: Hossain, G.M.S., Rashid, Md.H.O., Islam, Md.R., Sarker, A. and Yasmin, Must.A. (2022) Towards Mining Public Opinion: An Attention-Based Long Short Term Memory Network Using Transfer Learning. *Journal of Computer and Communications*, **10**, 112-131.
https://doi.org/10.4236/jcc.2022.106010

Received: April 25, 2022
Accepted: June 27, 2022
Published: June 30, 2022

Abstract

The Internet provides a large number of tools and resources, such as social media sites, online newsgroups, blogs, electronic forums, virtual communities, and online travel sites, for consumers to express their views or opinions regarding various issues. These opinions can help organizations like tourism to improve their products and services for their consumers. Opinion mining refers to a process of identifying emotions by applying Natural Language Processing (NLP) techniques to a pool of texts. This paper mainly focuses on mining public opinion from the hotel reviews domain. To do so, we proposed a novel technique called the Attention-Based Long Short Term Memory (Attention-LSTM) Network using a transfer learning approach. We empirically analyzed several machine learning and deep learning methods and observed our proposed technique provided an adequate performance for mining public opinion in the hotel reviews domain.

Keywords

Opinion Mining, Deep Learning, Word2Vec, Attention-LSTM, Transfer Learning

1. Introduction

Mining public opinions can be a tricky problem as there are a vast number of reviews available online. For instance, various travel sites like (TripAdvisor.com) and (Booking.com) contain a huge number of travel reviews, scores, ratings, and feedback. However, these online reviews help consumers to shape their travel

experiences and represent electronic word-of-mouth (eWoM). There is a report which indicates that 95% of customers before making their online hotel bookings browse online hotel reviews [1]. "Previous studies also confirmed the impact of online hotel reviews on consumers and the hotel industry as well" [2]. Furthermore, the consistency of the quality of reviews is another important issue. Several reviews contain biased information or are simply pointless, while on the contrary, other reviews are very helpful in objective evaluation. As a result, a huge number of reviews are explored by consumers who devote their adequate mental energy to reaching a specific opinion. Performing such an extensive study will certainly cost the consumers precious time. So, developing an efficient method for processing a large number of online reviews would be quite beneficial.

To do that, previous researchers applied a variety of Machine Learning (ML) and Deep Learning (DL) based techniques for classifying online hotel reviews. For instance, a supervised machine learning method was proposed for classifying hotel reviews in the work [3]. The research was conducted by applying Support Vector Machine (SVM) using TF-IDF features and Bag of Words (BOW). Logistic Regression (LR) and Naive Bayes (NB) Machine Learning (ML) approaches were applied in [4], for textual data analysis. Support Vector Classifier (SVC) technique was used by the author in [5] to classify the textual data accurately. However, data sparsity is a concerning issue for these models. On the other hand, Deep Learning (DL) techniques have gained immense popularity because of the lower feature engineering and expressive power of computations in NLP tasks than traditional models.

For effectively mining public opinion, especially from a domain like hotel reviews, creating a large corpus from a huge number of reviews and using that corpus to build a new corpus consisting of a small number of reviews can reduce the computational time and improve the accuracy. Because once the large corpus is developed then it can be reused to train other corpora which are comparatively small in size in less computational time. Hence the accuracy can be improved. In this paper, we implemented the above technique to build an effective corpus for hotel reviews classification.

The key contribution of this paper is to mine public opinion from the hotel reviews domain. To accomplish our objective, first, we developed word vectors using the Word2Vec model from an existing hotel reviews dataset, and then applied a transfer learning technique to develop word vectors for our gathered hotel reviews dataset. Secondly, we proposed an Attention-based Long Short Term Memory (Attention-LSTM) network for categorizing positive and negative opinions. And finally, we analyzed the performance of several Deep Neural Network (DNN) based models, such as LSTM, BiLSTM, GRU, BiGRU, and a hybrid architecture of CNN-LSTM with our proposed Attention-LSTM model for mining public opinion in the hotel reviews domain.

The rest of the paper is organized as follows. In Section 2, a brief overview of the related works in the hotel reviews classification domain is presented. In Sec-

tion 3, the materials and methodology of this paper are described. The results and discussion are explained in Section 4. We conclude this paper finally in Section 5.

2. Related Works

For mining public opinion, especially from hotel reviews, a significant amount of research has been performed over the years. A Convolutional Neural Network (CNN) based model for feature-based opinion mining from customer reviews in the hotel domain was developed in [6]. The authors obtained 98.22% accuracy for combined reviews, and 95.345% and 96.145% accuracy for the positive and negative reviews, respectively. A Fuzzy domain ontology combined with Support Vector Machine (SVM) was applied to automate the online review classification in the work [7] and achieved an accuracy of 82.7%. Several machine learning-based techniques such as Naive Bayes, Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), etc., were used for sentiment analysis or mining opinions in the works [7] [8] [9] [10] [11]. SentiWordNet, which is derived from the WordNet database, is a widely used technique for scoring the positivity or negativity of the words to classify the reviews.

A SentiWordNet based model was proposed by the authors in [9] and got 87% accuracy in classifying the positive and negative reviews from hotel reviews. Visual analytics along with a multi-feature fusion CNN model can also be applied to classify the customers' responses. To empirically identify managerial responses, the authors in [12] are among the first to develop such a model. They used computational linguistics, visual analytics along with a multi-feature fusion CNN model to analyze hotel reviews and identify response strategies.

Using both lexical and word vectors methods to analyze words spherically, the authors in the work [13] found a better result in terms of reduced computation time for mining opinions. A text summarization approach using the k-mediods clustering algorithm was developed in [14] to take into consideration some crucial issues such as author credibility and conflicting reviews in the opinion mining problem.

To classify praise or complaint using linguistic-based hybrid features of extreme opinions, the authors in [15] compared Machine Learning, Ensemble, and Deep Neural Network-Based methods. They achieved an f1-score of 96.23% for multichannel CNN and an f1-measure of 99.7% for ensemble algorithm. A deep learning-based model using word embedding and Gated Recurrent Unit (GRU) that can automatically perform hotel reviews classification was introduced in [16] and provided an accuracy of 89% with 92% fi-score.

Another widely used Deep Neural Network (DNN), LSTM-RNN was implemented by the authors in the work [17]. They evaluated the model on a large dataset of hotel reviews with word embedding features. They got an accuracy of 97% and 76.53% of f1-score and claimed the effectiveness of the model on any review classification-based tasks [17]. An NLP platform, OpeNER, was applied

to the hospitality domain for processing customer reviews and to obtain valuable information developed in [18]. The platform has a set of free NLP tools to process the textual content on a modular architecture. For training and evaluating the platform, a manually annotated hotel reviews dataset was used. However, most of these works do not use any pretrained corpus to generate more accurate word vectors. This paper firstly creates a corpus only for the hotel reviews domain and uses this built corpus to generate more accurate word vectors for the experimental dataset and used a novel technique Attention-LSTM for classifying them into positive or negative categories. In the next section, we will discuss the materials and methodology used to conduct this research.

3. Materials and Methodology

3.1. Dataset Description

In this research, we used two separate datasets to carry out our experiments. Firstly, we collected a dataset from Kaggle which contains customer reviews of 515 K hotels in Europe [19]. This dataset has 17 fields. From which we used only two, namely "Positive_Review" and "Negative_Review" as our main intention was to mine opinions from the customer reviews. The dataset consists of an equal number (515,738 reviews) of positive and negative reviews. In the following **Table 1**, some reviews along with the opinion category of this dataset are shown.

We developed another dataset by gathering around 1.5 K reviews (Bangladeshi Hotels) from (Booking.com) mainly to conduct various analyses to mine public opinion. The second dataset contains 3 attributes from which we took 2 attributes, namely "Review" and "Sentiment". The "Review" field contains both positive and negative reviews. The positive reviews are labeled with 1 whereas the negative ones are labeled with 0. The dataset has 1042 positive reviews and 457 negative reviews. **Table 2** shows some examples of customer praise and complaints about various hotels. The most common words of this dataset are represented in the wordcloud at **Figure 1**. In the next part, we will discuss the proposed methodology used in this research.

Table 1. Sample reviews from 515 K hotel reviews dataset.

#	Review	Class
1	The aircondition makes so much noise and its hard to sleep at night.	Negative
2	Comfy bed good location.	Positive
3	Transportation was a bit of a pain but onroute to your destination there is amazing views at every corner.	Negative
4	Great hotel original concept style.	Positive
5	Not cleaned well lady pushing to pay during my breakfast poor signs for temporary reception during renovation.	Negative

Figure 1. Most common words from experimental dataset (Booking.com).

Table 2. Sample reviews from experimental dataset (Booking.com).

#	Review	Sentiment
1	Exceptional. Near Sea Beach.	1
2	Poor. Room is not good to stay. Service is disgusting. There is no privacy. No personal balcony. Bathroom condition is bogus.	0
3	Very Good. Location, Restaurant Room Service.	1
4	Its very disappointing. You must have enough money to spend a night there. It is overpriced. The food is overrated. Dinner is expensive. We two people paid 3000 taka for dinner buffet. There are not enough items.	0
5	Hotel was so bad and for this I wasn't able to enjoy my trip	0

3.2. Proposed Methodology

Figure 2 indicates the methodology used in this research. The first layer of our proposed framework is corpus building for our experimental dataset, which includes text preprocessing, *i.e.* tokenization, removal of punctuations, non-alphabetic token removal, and stopwords removal. Initially, we preprocessed the 515 K hotel reviews dataset and developed a corpus using the word embedding technique word2vec. In the following, we preprocessed the experimental dataset (Booking.com). The corpus developed in the earlier step was then used to train the preprocessed text of the experimental dataset (Booking.com) to build a corpus. This process is called "transfer learning" because here, a pretrained large corpus was used to train a relatively small corpus and helped to generate strong word vectors that could not be found if we developed word vectors directly using word2vec.

The Attention-Based Long Short Term Memory (Attention-LSTM) model was introduced in the next step. The model was implemented on the experimental

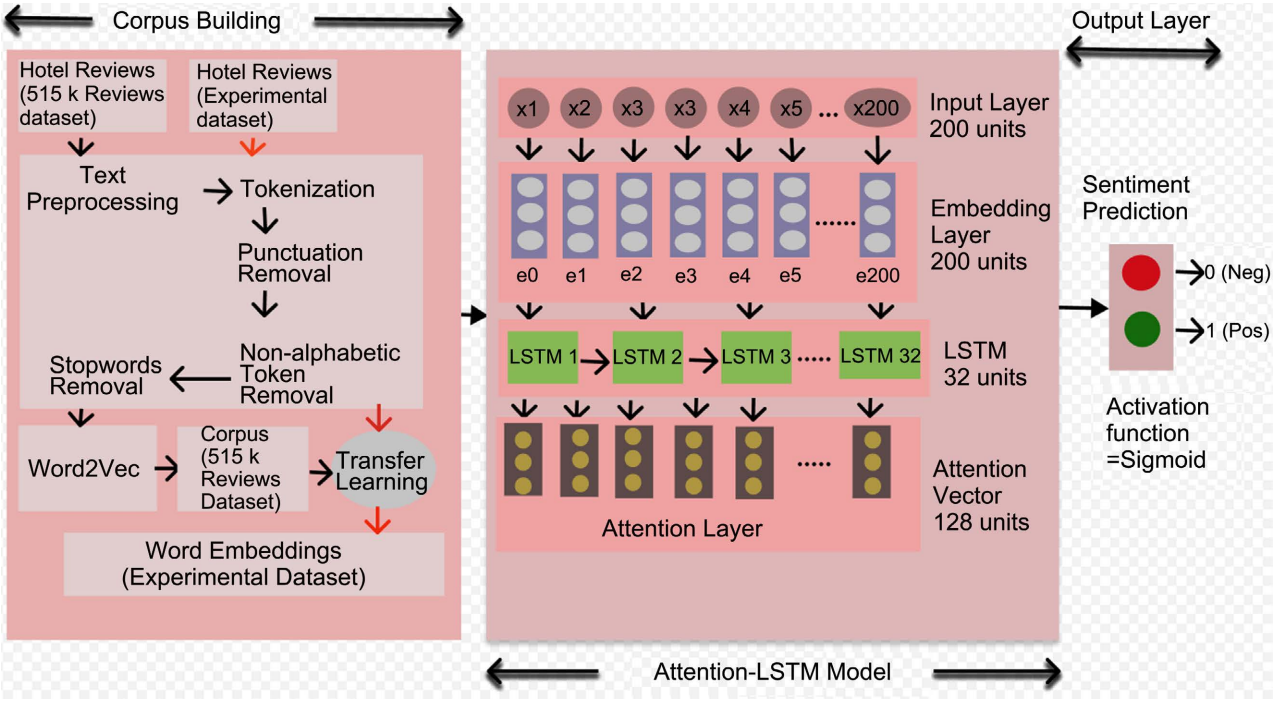

Figure 2. Proposed methodology based on Attention-LSTM model.

dataset to mine public opinion. Finally, the prediction was measured for the positive or negative opinion in the output layer. In the remaining subsections, details of our methodology are described.

3.2.1. Text Preprocessing

Text preprocessing means cleaning text data by removing the noise and making text data ready to feed into machine learning models. In the actual scenario, text data is mixed up with punctuation, stop words, emoticons, and non-alphabetic elements. Such types of noise must be removed before further processing. In this research, we first conducted text preprocessing for both datasets to remove unnecessary elements. Text preprocessing is done by the following steps:

- Tokenization refers to the process of extracting the smaller units called tokens from a piece of text. Tokens can be made of characters, words, or subwords. For example, if a hotel review is like "the hotel staff were very friendly", after tokenization, we will get tokens such as "the", "hotel", "staff", "were", "very", "friendly". We applied tokenization to each sentence of our datasets and generated tokens.
- A punctuation mark can be a mark or character used for separating sentences or phrases. Common punctuation marks used are period(.), comma(,), semicolon(;), question mark(?), or dash(-) etc. For further text processing, we removed punctuations from each token as punctuation marks do not play a significant role in the case of text processing.
- Most of the reviews consist of some non-alphabetic tokens such as emojis, emoticons, or symbols etc. These tokens need to be removed for text processing

as there will not be any huge impact on the classification of reviews.

- Stop words are words that provide no useful information for determining which category a text should be classified in. This could be because they have no meaning (prepositions, conjunctions, etc.) or because they are overused in the classification context. So the stop words like "a", "the", "in" etc., are removed from the token list at the end of the text preprocessing step.

3.2.2. Word2Vec

Word2Vec is a neural network consisting of one hidden layer and has weights. During the training, the model uses a back-propagation technique to adjust those weights to reduce the loss function. Word2Vec model takes only the hidden weights which are the word embeddings or vectors after the training is completed. Preprocessed text data generated from the previous steps is used for producing word embeddings. To do so, the preprocessed texts of the 515 K hotel reviews dataset were fed into the Word2Vec model. In this paper, we took vector_size = 200, window = 5 and min_count = 1 as parameters in our Word2Vec model. We saved the word embeddings of the 515 K hotel reviews dataset and later used them to generate word embeddings for our gathered dataset (Booking.com). Table 3 shows the top 10 most similar words and their probability score for the words "room", "staff", and "airconditioner" respectively. Similar words are found after the training of the experimental dataset using a transfer learning technique, and it can be seen that word predictions tend to be more accurate.

In our gathered dataset, we had around 1.5 K reviews, as described earlier, among them 1042 positive reviews and 457 negative reviews. As there is an imbalance between positive and negative reviews, we performed oversampling at

Table 3. Top 10 most similar words with probability score from experimental dataset after transfer learning.

#	Word		Word		Word	
	"room"	Score	"staff"	Score	"airconditioner"	Score
1	rooms	0.754	staffs	0.671	aircondition	0.849
2	bedroom	0.646	personnel	0.642	airconditioning	0.835
3	originally	0.472	receptionists	0.607	airco	0.802
4	bed	0.466	receptionist	0.572	ac	0.780
5	also	0.465	employees	0.561	aircon	0.779
6	suite	0.458	stuff	0.560	c	0.763
7	bathroom	0.455	team	0.556	thermostat	0.719
8	initially	0.451	manner	0.517	thermostats	0.699
9	double	0.438	lady	0.507	regulator	0.661
10	allocated	0.428	gentleman	0.491	heating	0.641

the very beginning. Padding was also performed because all the reviews in our dataset did not have the same sentence length. Padding is a method that is used to maintain the same input size for machine learning or deep learning models. All the models operated on the same input length. That's why padding was necessary. We performed padding by taking a maximum length of 200. In the following subsection, we introduce the proposed Attention-LSTM model.

3.2.3. Attention-LSTM

To mine public opinion, we introduced the Attention-LSTM model, which is summarized in **Figure 3**. The architecture takes advantage of the sparsity of the word embedding matrix. The word embedding matrix is the vector representation of all textual comments carrying positive and negative sentiment. In our case, the dimension of the embedding matrix was 200. We developed the embedding matrix in such a way that the effect of the curse of dimensionality becomes negligible. The first layer of our architecture was an input layer of 200 units, which is expressed as [x1, x2, x3, …, x200] where x represents the input features of each review bearing positive or negative sentiment. The input features are nothing, but the word vectors stored in the embedding matrix. The following layer of our architecture was the embedding layer of shape (200, 200) denoted as [e0, e1, e3, …, e200] as shown in **Figure 2**. To preserve consistency, we kept the same shape for the embedding layer as the input layer. The output of the embedding layer was then provided as the input to the next LSTM layer, which had 32 units of LSTM.

summary of the built model...
Model: "sequential 27"

Layer (type)	Output Shape	Param #
embedding_27 (Embedding)	(None, 200, 200)	543800
lstm_23 (LSTM)	(None, 200, 32)	29824
last_hidden_state (Lambda)	(None, 32)	0
attention_score_vec (Dense)	(None, 200, 32)	1024
attention_score (Dot)	(None, 200)	0
attention_weight (Activation)	(None, 200)	0
context_vector (Dot)	(None, 32)	0
attention_output (Concatenate)	(None, 64)	0
attention_vector (Dense)	(None, 128)	8192
dense_24 (Dense)	(None, 1)	129

Total params: 582,969
Trainable params: 39,169
Non-trainable params: 543,800

Figure 3. Summary of attention-LSTM model architecture.

Long Short Term Memory (LSTM) is a type of recurrent neural network that tries to remember all the previous knowledge that the network has seen so far and forgets irrelevant data. The memory, cell c_t of the LSTM network is able to remember the previous states over very long periods, removing the dependency problem of RNN [20]. This memory cell is the core of the LSTM network and is recurrently connected to itself. LSTM has three gates, namely input i_t, forget f_t, and output o_t gate, respectively, as shown in **Figure 4**, which knowledge needs to be saved or forgotten is decided by the cell using the gating mechanism.

Consider tanh(.), σ(.), and $\otimes$ are the hyperbolic tangent function, element-wise sigmoid function, and product, respectively. Suppose h_t and x_t are the hidden state vector and the input vector at time t. W contains the weight matrices of the hidden state h_t and U contains the weight matrices of the input x_t and bias vectors are denoted by b. The forget gate of an LSTM cell then works based on the following "Equation (1)" to decide what needs to be forgotten [21].

The input gate computes i_t and $\tilde{c_t}$ and combine them according to the following Equation (1), Equation (2), Equation (3), and Equation (4) to decide what new data needs to be stored.

$$f_t = \sigma\left(W_f h_{t-1} + U_f x_t + b_f\right) \tag{1}$$

$$i_t = \sigma\left(W_i h_{t-1} + U_i x_t + b_i\right) \tag{2}$$

$$\tilde{c_t} = \tanh\left(W_c h_{t-1} + U_c x_t + b_c\right) \tag{3}$$

$$c_t = f_t \otimes c_{t-1} + i_t \otimes \tilde{c_t} \tag{4}$$

The output gate represents the output by selecting the particular parts of cell state based on the below equations

$$o_t = \sigma\left(W_o h_{t-1} + U_o x_t + b_o\right) \tag{5}$$

$$h_t = o_t \otimes \tanh\left(c_t\right) \tag{6}$$

The output of the LSTM layer is then sent to the attention layer, which is a crucial component of our architecture for further processing. An attention mechanism was applied to solve the problem of long-distance dependency from the

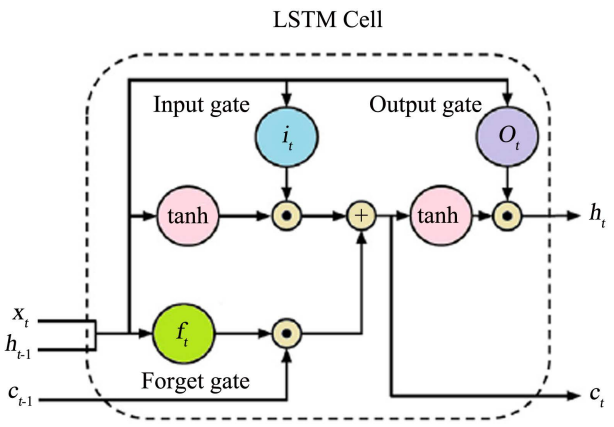

Figure 4. Sample architecture of LSTM [21].

experimental dataset. The idea behind the attention mechanism is that to infer the sentiment of a review, all aspects do not necessarily need to be considered, rather needs to focus on important aspects of a review. The Attention layer does that by utilizing some weight on the input data [22]. An additive attention mechanism was applied in our architecture. The output of our attention layer was a vector of 128 dimensions, which was fed to the last layer of the architecture. The final layer of our architecture had a single unit neuron with a sigmoid activation function and was responsible for outputting positive or negative opinions. Equation (7) defines the sigmoid activation function.

$$\sigma(z) = 1/\left(1+\mathrm{e}^{-z}\right) \tag{7}$$

where z is the input variable and $\sigma(z)$ is the sigmoid activation function with a range of [0, 1]. In the following section, the model's performance and the experimental results are explained.

4. Results and Discussion

4.1. Model Compilation and Evaluation

Once the model was built, the next step was to compile the model. For compiling the model, we used "binary_crossentropy" as a loss function, "adam" as an optimizer, and "accuracy" as metrics., If y_i is the target value, p_i is the predicted value, and N is the number of output values, then binary cross-entropy or log loss can be measured by using Equation (8) [23] stated below.

$$\log\text{loss} = \frac{1}{N}\sum_{i=1}^{N} -\left(y_i \log p_i + (1-y_i)\log(1-p_i)\right) \tag{8}$$

Model evaluation was performed in terms of accuracy, precision, recall, and f1-score. Accuracy can be defined as the percentage of accurate predictions for the test data. It can be measured by dividing the number of accurate predictions by the number of overall predictions.

$$\text{Accuracy} = \text{Accurate Predictions/Overall Predictions}$$

Precision can be defined as the fraction of true positives and the sum of true positives and false positives.

$$\text{Precision} = \text{True Positives/True Positives} + \text{False Positives}$$

Recall can be defined as the fraction of true positives and the sum of true positives and false negatives.

$$\text{Recall} = \text{True Positives/True Positives} + \text{False Negatives}$$

F1-score is a function of precision and recall.

$$\text{F1-score} = 2*\left(\text{Precision}*\text{Recall}\right)/\left(\text{Precision}+\text{Recall}\right)$$

The deep learning models were implemented using the Keras library, which is a high-level API of TensorFlow and is widely used for solving machine learning problems. We used an Intel (R) core (TM) i5-10300H CPU with 16 GB of RAM and an Nvidia GTX 1650 GPU platform to carry out our experiments. We split

our gathered hotel reviews dataset (Booking.com) into the train, validation, and test sets. We used 70% of our dataset for training, 10% as validation, and 20% for testing our models. The models were executed for 50 epochs with batch_size = 128 and to avoid the overfitting problem we also used a dropout layer of 20%.

4.2. Performance Analysis

Table 4 shows the performance of the various machine and deep learning techniques used in this paper. Precision, recall, f1-score, and accuracy are used for measuring the performance of the techniques. From Table 4 it can be seen that deep learning methods performed better than machine learning methods on our experimental dataset. Among the machine learning methods, we found Decision Tree as the best technique, followed by Random Forest, SVC, and Multinomial Naive Bayes. The Decision Tree obtained an accuracy of 90% with an 88.6% and 94% precision score for mining negative and positive reviews, respectively.

Table 4. Results obtained on the experimental dataset (Booking.com).

Method	Class	Precision	Recall	F1-Score	Accuracy
Decision Tree	Negative	0.88	0.95	0.91	0.90
	Positive	0.94	0.85	0.89	
SVC	Negative	0.77	0.49	0.60	0.65
	Positive	0.60	0.83	0.69	
Random Forest	Negative	0.86	0.94	0.90	0.89
	Positive	0.92	0.82	0.87	
Multinomial Naïve Bayes	Negative	0.62	0.03	0.05	0.48
	Positive	0.47	0.98	0.64	
LSTM (LR = 0.001)	Negative	0.97	0.95	0.96	0.9599
	Positive	0.95	0.97	0.96	
BiLSTM (LR = 0.001)	Negative	0.96	0.95	0.96	0.9573
	Positive	0.95	0.96	0.96	
GRU (LR = 0.001)	Negative	0.97	0.96	0.97	0.9563
	Positive	0.96	0.97	0.96	
BiGRU (LR = 0.001)	Negative	0.96	0.95	0.96	0.9553
	Positive	0.95	0.96	0.96	
CNN-LSTM (LR = 0.001)	Negative	0.97	0.96	0.97	0.9679
	Positive	0.96	0.97	0.97	
Attention-LSTM (LR = 0.01)	Negative	**0.97**	**0.97**	**0.97**	**0.9706**
	Positive	**0.97**	**0.97**	**0.97**	

From the deep learning models, our proposed Attention-LSTM provided the highest performance with 97% precision, recall, and f1-score and outperformed others by acquiring 97.06% accuracy. Two specific steps worked well for the Attention-LSTM model. Firstly, the well-trained word vectors that we achieved by using the transfer learning technique and, secondly, the attention layer we introduced at the end of our architecture. The attention layer assigned some random weights to acquire more accurate word vectors and helped remember the word sequence in a sentence and to categorize positive or negative opinions. We kept the model lightweight as much as possible and saw that it took approximately 3 seconds to complete 1 epoch during a training session. We executed our proposed architecture for several learning rates (LRs), such as with LR = 0.001, 0.002, 0.003, 0.004, 0.008, and for 0.01. At a learning rate of 0.001, the Attention-LSTM model worked best for categorizing positive and negative opinions.

On the other hand, the CNN-LSTM model is slightly behind in terms of performance from our recommended architecture, with an accuracy of 96.79%. Several findings can be mentioned regarding the performance of the CNN-LSTM model. Firstly, using the pretrained word vectors to develop finely tuned word vectors as mentioned earlier. Secondly, the 1-dimensional convolutional layer with 32 filters and a kernel size = 3 extracted the features well and sent them to the LSTM layer for classifying the positive and negative opinions. The CNN-LSTM architecture was implemented with a learning rate of 0.001 and it took approximately 4 seconds to complete the first epoch. We carried through a few experiments by employing variations of LSTM on our dataset. Both LSTM and Bidirectional LSTM (BiLSTM) provided the same performance, while GRU and BiGRU produced approximately equal performance. All of them were executed with a learning rate of 0.001.

Table 5 depicts the confusion matrix for all of the techniques used in this research. We found that our proposed Attention-LSTM model gave 1.33% of false-negative predictions and 1.60% of false-positive predictions. Besides, 51.20% of true negative and 45.87% of true positive predictions were made while classifying the reviews on the test dataset. While working on the test dataset, we observed the most false-positive output of 51.47% for Multinomial Naive Bayes classifier and the most false-negative output of 8.27% for the Random Forest classifier.

Figure 5 and **Figure 6** depict the training and validation accuracy of our proposed Attention-LSTM model for the different learning rates (LRs) of the optimizer. As the learning rates are close in terms of their values, as we observe in **Figure 5**, there is not too much of a significant difference in the training accuracy. This is because of close learning rates (LR = 0.001, 0.002, 0.003, 0.004, 0.008, and 0.01). We observe in **Figure 5**, if the learning rate closely increases, the model learns faster and provides almost similar performance in the training period. In **Figure 6**, we notice some spikes in the validation accuracy for various learning rates. This is perhaps because of the 20% dropout layer after each epoch

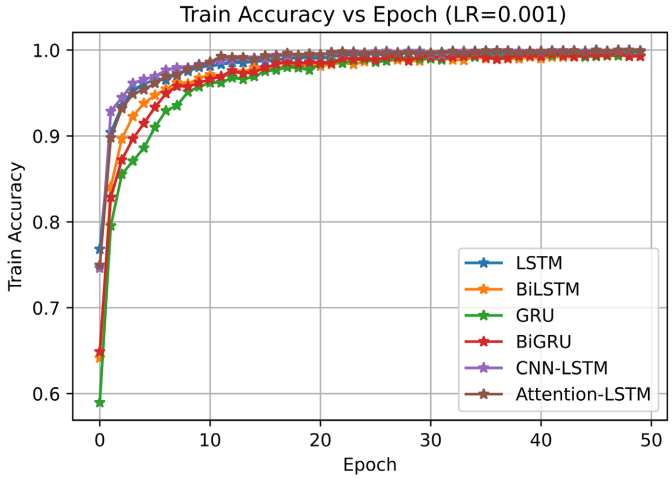

Figure 5. Training accuracy of attention-LSTM with a varying number of learning rates (LRs) of the optimizer. The X-axis denotes the epochs, whereas the Y-axis denotes the training accuracy within a range of 0 and 1. Epoch means training the model with the training dataset once. The higher accuracy with fewer epochs is considered better performance for the model.

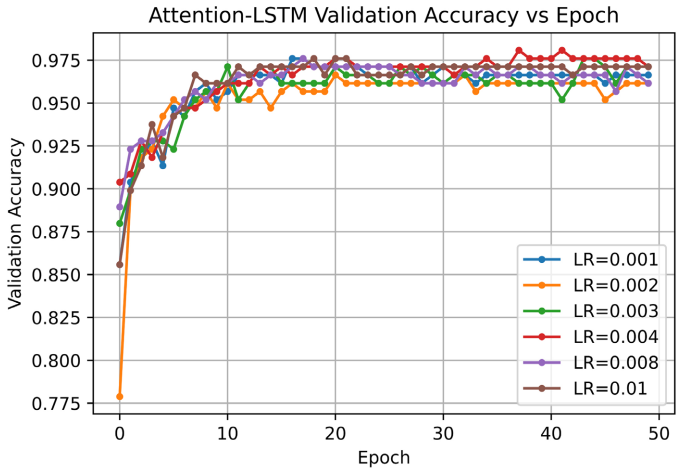

Figure 6. Validation accuracy of Attention-LSTM with a varying number of learning rates (LRs) of the optimizer. The X-axis denotes the epochs, whereas the Y-axis denotes the validation accuracy within a range of 0 and 1. Data in the validation set is independent of the training set, and validation accuracy indicates how well the model performs for unseen data.

used in our Attention-LSTM architecture. **Figure 7** and **Figure 8** represent the training and validation loss of the Attention-LSTM architecture. In **Figure 7**, we have seen that training loss gradually decreases when epoch increases for various learning rates as mentioned in **Figure 5**. We plotted the training and validation accuracy of the Attention-LSTM model together in **Figure 9** to determine whether there is any overfitting issue or not. From **Figure 9**, we observe that there is a marginal distance between training and validation accuracy. **Figure 10** shows the train and validation loss combined for various LRs of the Attention-LSTM model.

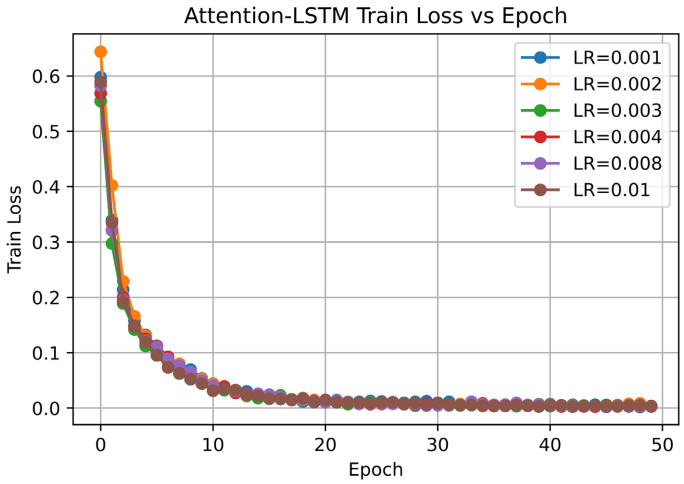

Figure 7. Training loss of attention-LSTM with a varying number of learning rates (LRs) of the optimizer. The X-axis denotes the epochs, whereas the Y-axis denotes the training loss. Binary cross-entropy was used for measuring the training loss. The lower the loss with fewer epochs, the better the performance of the model.

Table 5. Confusion matrix obtained on the experimental dataset (Booking.com).

Method	Actual	Predicted	
		Negative	Positive
Decision Tree	Negative	50.40%	2.40%
	Positive	7.20%	40%
SVC	Negative	26.13%	26.67%
	Positive	8%	39.20%
Random Forest	Negative	49.60%	3.20%
	Positive	8.27%	38.93%
Multinomial Naïve Bayes	Negative	1.33%	51.47%
	Positive	0.80%	46.40%
LSTM (LR = 0.001)	Negative	50.40%	2.40%
	Positive	1.60%	45.60%
BiLSTM (LR = 0.001)	Negative	50.40%	2,40%
	Positive	1.87%	45.33%
GRU (LR = 0.001)	Negative	51.20%	1.60%
	Positive	1.60%	45.60%
BiGRU (LR = 0.001)	Negative	50.93%	1.87%
	Positive	1.60%	45.60%
CNN-LSTM (LR = 0.001)	Negative	50.93%	1.87%
	Positive	1.33%	45.87%
Attention-LSTM (LR = 0.01)	Negative	**51.20%**	**1.60%**
	Positive	**1.33%**	**45.87%**

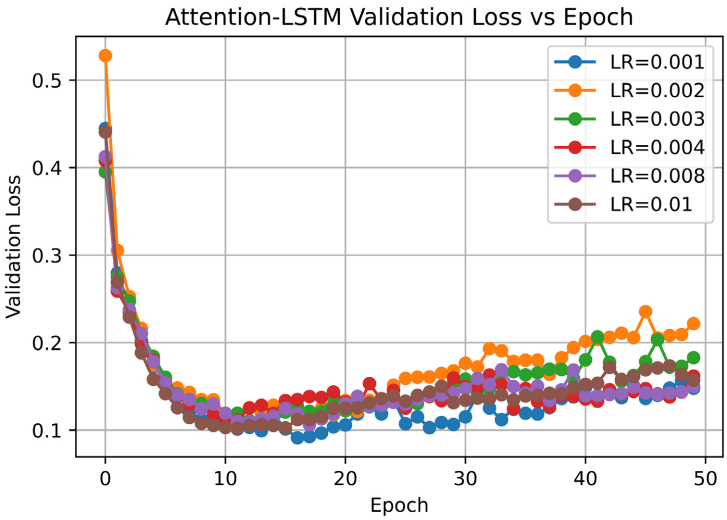

Figure 8. Validation loss of attention-LSTM with a varying number of learning rates (LRs) of the optimizer. The X-axis denotes the epochs, whereas the Y-axis denotes the validation loss. Validation loss was measured from the validation set. The lower the loss with fewer epochs, the better the model's performance for future unseen data.

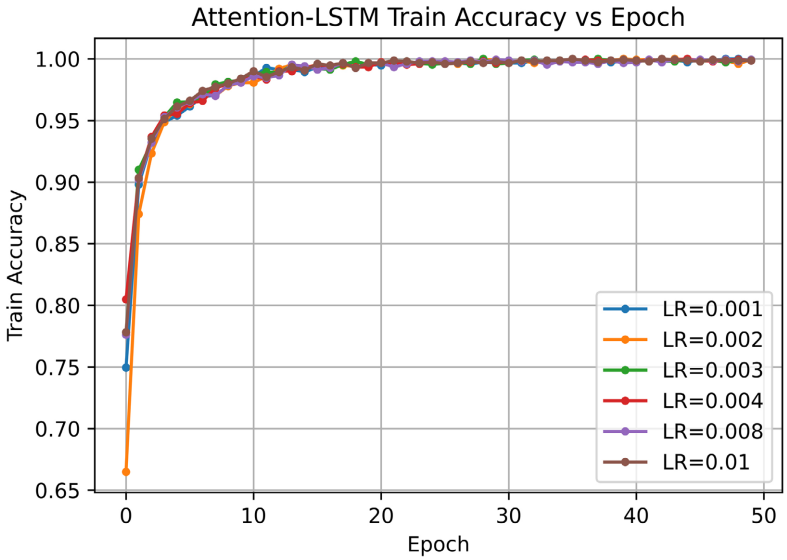

Figure 9. Plotting attention-LSTM's training and validation accuracy together where TA represents the training accuracy and VA represents the validation accuracy for various learning rates (LRs) of the optimizer.

To compare the performance of our proposed model during training and validation sessions with various deep learning techniques, we drew the graphs shown in **Figures 11-14**. The first two denote the training and validation accuracy for LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and Attention-LSTM, whereas the last two represent the training and validation loss respectively. In **Figure 11** we observe some deviant behavior for Attention-LSTM model between epoch 1 and 20 during training period. This is maybe because of the random weights initialized by the model itself at certain epochs. In **Figure 13**, we also notice that our

proposed model achieves less training loss comparatively. Although the performance of the used deep learning techniques looks similar to the graphs, the Attention-LSTM model is slightly ahead of all of the methods used in this research.

Table 6 shows the performance comparison of our proposed model with a few state-of-the-art methods used for public sentiment analysis from the labelled hotel reviews dataset [24]. Our proposed Attention-LSTM model outperforms others by achieving an accuracy of 92% with 92% F1-Score.

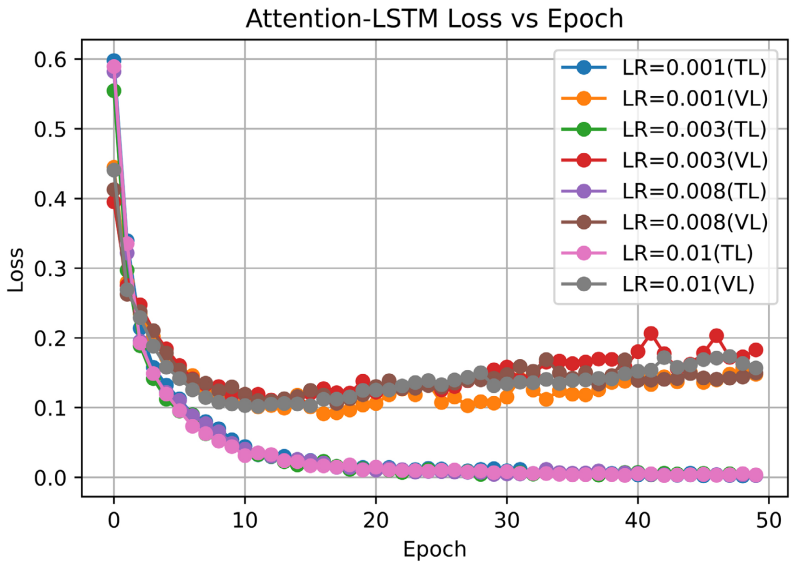

Figure 10. Plotting attention-LSTM's training and validation loss together where TL represents the training loss and VL represents the validation loss for various learning rates (LRs) of the optimizer.

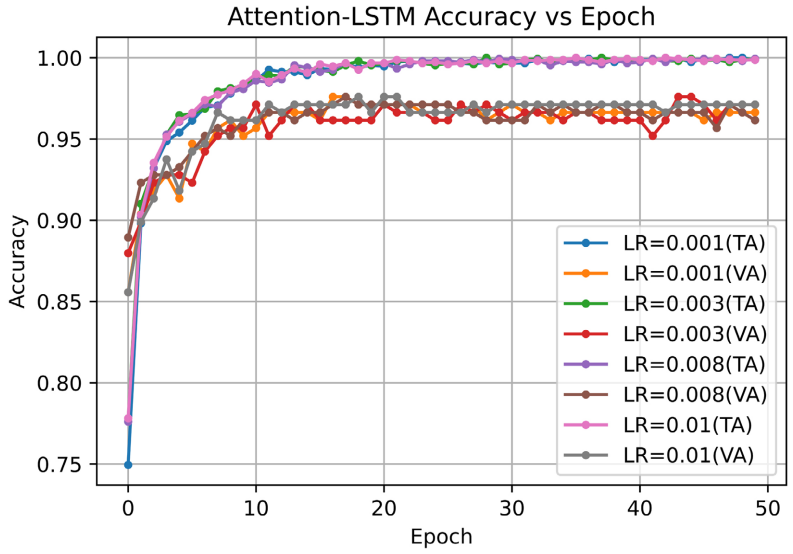

Figure 11. Training accuracy of LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and Attention-LSTM for learning rate (LR) = 0.001 of the optimizer. Training accuracy was measured from the same training set for all of the methods used here.

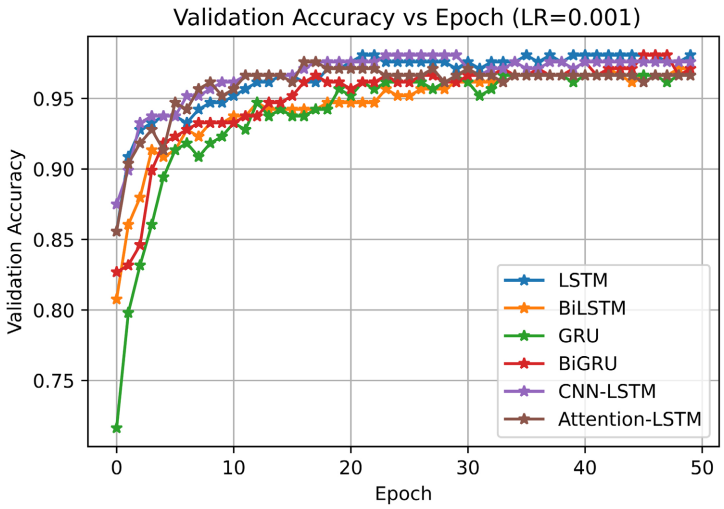

Figure 12. Validation accuracy of LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and Attention-LSTM for learning rate (LR) = 0.001 of the optimizer. Validation accuracy was measured from the same validation set for all of the methods used here.

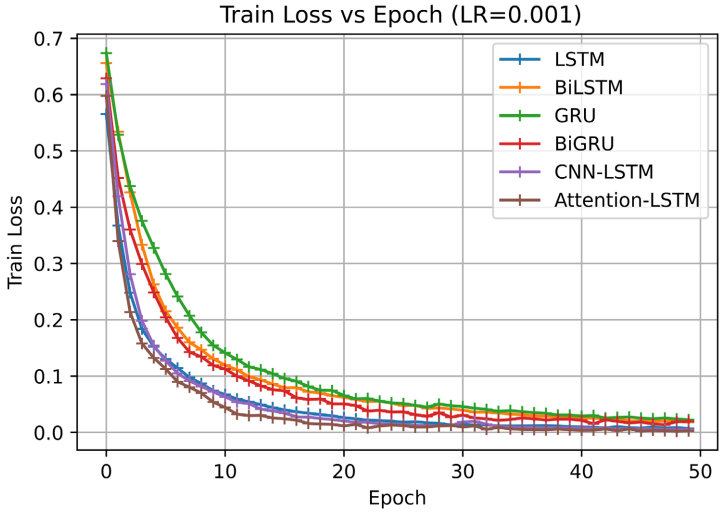

Figure 13. Training loss of LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and Attention-LSTM for learning rate (LR) = 0.001 of the optimizer. Training loss was measured from the same training set and using binary cross-entropy loss for all of the methods used here.

Table 6. Performance comparison of proposed Attention-LSTM model on labelled hotel reviews dataset [24].

Dataset [24]	Used Method	Accuracy	F1-Score	Reference
Labelled Hotel Reviews	Gated Recurrent Unit (GRU)	89%	92%	Anis S. *et al.* [16]
Labelled Hotel Reviews	Fuzzy Cardinality AFINN Approach	76.2%	-	S. Vashishtha and S. Susan [25]
Labelled Hotel Reviews	**Attention-LSTM**	92%	92%	This Paper

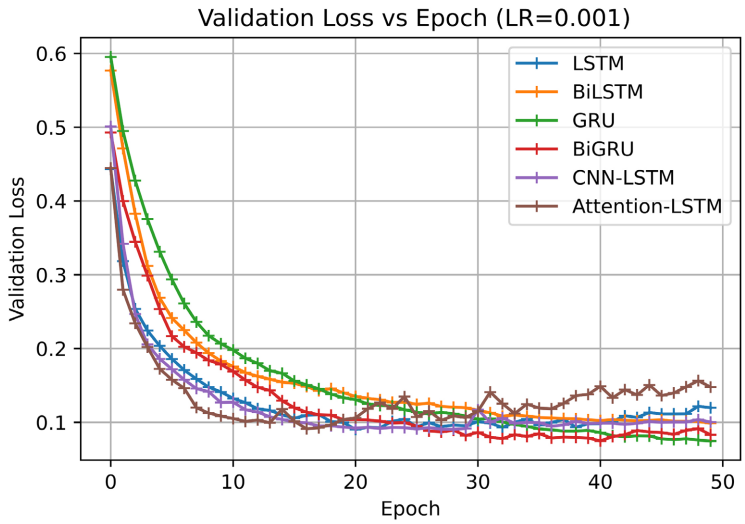

Figure 14. Validation loss of LSTM, BiLSTM, GRU, BiGRU, CNN-LSTM, and Attention-LSTM for learning rate (LR) = 0.001 of the optimizer. Validation loss was measured from the same validation set and using binary cross-entropy loss for all of the methods used here.

5. Conclusion

Every day on the web, a large amount of consumer-generated textual content is appearing and creating a huge challenge and a big opportunity. Specialized websites like (TripAdvisor.com) and (Booking.com) allow consumers to write reviews and publish their opinions, clearly impacting the hotel domain. As a result, mining public opinion from consumer-generated reviews will surely contribute to tourism organizations and the consumer's well-being. In this paper, we concentrated on mining public opinion from the hotel reviews domain and proposed a novel framework, Attention-LSTM, to attain the objective of our study. We implemented several Deep Learning (DL) approaches such as LSTM, BiLSTM, GRU, BiGRU, and a hybrid architecture of CNN-LSTM, and analyzed the performance with our recommended model. Initially, we used an existing 515 K hotel reviews dataset (kaggle) to build word embeddings and then applied the transfer learning technique to develop word embeddings for our gathered hotel reviews dataset (Booking.com). We found that the Attention-LSTM model performs better than other approaches by achieving 97.06% accuracy and provides an up-to-the-mark result compared with the state-of-the-art techniques. In the future, we will apply several other datasets to justify the performance of our proposed architecture and move towards aspect-based opinion mining.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Ady, M. and Quadri-Felitti, D. (2015) Consumer Research Identifies How to Present

Travel Review Content for More Bookings. *Hotels News Resource*, 95.

[2] Ghose, A. and Ipeirotis, P.G. (2011) Estimating the Helpfulness and Economic Impact of Product Reviews: Mining Text and Reviewer Characteristics. *IEEE Transactions on Knowledge and Data Engineering*, **23**, 1498-1512. https://doi.org/10.1109/TKDE.2010.188

[3] Shi, H. and Li, X. (2011) A Sentiment Analysis Model for Hotel Reviews Based on Supervised Learning. 2011 *International Conference on Machine Learning and Cybernetics*, Guilin, 10-13 July 2011, 950-954. https://doi.org/10.1109/ICMLC.2011.6016866

[4] Mccallum, A. and Nigam, K. (1998) A Comparison of Event Models for Naive Bayes Text Classification. 1998 *AAAI Workshop*, Madison, 26-27 July 1998, 41-48.

[5] Joachims, T. (1999) Making Large Scale SVM Learning Practical. In: Scholkopf, B., Burges, C. and Smola, A., Eds., *Advances in Kernel Methods*, MIT Press, Cambridge, 169-184.

[6] Lal, K. and Mishra, N. (2020) Feature Based Opinion Mining on Hotel Reviews Using Deep Learning. In: Raj, J., Bashar, A. and Ramson, S., Eds., *Innovative Data Communication Technologies and Application ICIDCA* 2019, Springer, Berlin, 616-625. https://doi.org/10.1007/978-3-030-38040-3_70

[7] Ali, F., Kwak, K.-S. and Kim, Y.-G. (2016) Opinion Mining Based on Fuzzy Domain Ontology and Support Vector Machine: A Proposal to Automate Online Review Classification. *Applied Soft Computing*, **47**, 235-250. https://doi.org/10.1016/j.asoc.2016.06.003

[8] Raut, V.B. and Londhe, D.D. (2014) Opinion Mining and Summarization of Hotel Reviews. 2014 *International Conference on Computational Intelligence and Communication Networks*, Toronto, 10-12 January 2014, 556-559. https://doi.org/10.1109/CICN.2014.126

[9] Puri, C., Yadav, A., Jangra, G., Saini, K. and Kumar, N. (2017) Opinion Mining from Social Travel Networks. In: Banati, H., Bhattacharyya, S., Mani, A. and Köppen, M., Eds., *Hybrid Intelligence for Social Networks*, Springer, Cham, 177-206. https://doi.org/10.1007/978-3-319-65139-2_8

[10] Lee, P.-J., Hu, Y.-H. and Lu, K.-T. (2018) Assessing the Helpfulness of Online Hotel Reviews: A Classification-Based Approach. *Telematics and Informatics*, **35**, 436-445. https://doi.org/10.1016/j.tele.2018.01.001

[11] Tsai, C.-F., Chen, K., Hu, Y.-H. and Chen, W.-K. (2020) Improving Text Summarization of Online Hotel Reviews with Review Helpfulness and Sentiment. *Tourism Management*, **80**, Article ID: 104122. https://doi.org/10.1016/j.tourman.2020.104122

[12] Chang, Y.-C., Ku, C.-H. and Chen, C.-H. (2020) Using Deep Learning and Visual Analytics to Explore Hotel Reviews and Responses. *Tourism Management*, **80**, Article ID: 104129. https://doi.org/10.1016/j.tourman.2020.104129

[13] Rizkallah, S., Atiya, A.F. and Shaheen, S. (2021) Learning Spherical Word Vectors for Opinion Mining and Applying on Hotel Reviews. In: Abraham, A., Piuri, V., Gandhi, N., Siarry, P., Kaklauskas, A. and Madureira, A., Eds., *Intelligent Systems Design and Applications. ISDA* 2020, Advances in Intelligent Systems and Computing, Vol. 1351, Springer, Cham, 200-211. https://doi.org/10.1007/978-3-030-71187-0_19

[14] Hu, Y.-H., Chen, Y.-L. and Chou, H.-L. (2017) Opinion Mining from Online Hotel Reviews—A Text Summarization Approach. *Information Processing & Management*, **53**, 436-449. https://doi.org/10.1016/j.ipm.2016.12.002

[15] Khedkar, S. and Shinde, S. (2020) Deep Learning and Ensemble Approach for Praise or Complaint Classification. *Procedia Computer Science*, **167**, 449-458. https://doi.org/10.1016/j.procs.2020.03.254

[16] Anis, S., Saad, S. and Aref, M. (2021) Deep Learning-Based Approach for Sentiment Classification of Hotel Reviews. In: Kumar, S., Purohit, S.D., Hiranwal, S., Prasad, M., Eds., *Proceedings of International Conference on Communication and Computational Technologies. Algorithms for Intelligent Systems*, Springer, Singapore, 211-218. https://doi.org/10.1007/978-981-16-3246-4_16

[17] Ishaq, A., Umer, M., Mushtaq, M.F., *et al.* (2021) Extensive Hotel Reviews Classification Using Long Short-Term Memory. *Journal of Ambient Intelligence and Humanized Computing*, **12**, 9375-9385. https://doi.org/10.1007/s12652-020-02654-z

[18] García-Pablos, A., Cuadros, M. and Linaza, M.T. (2016) Automatic Analysis of Textual Hotel Reviews. *Information Technology & Tourism*, **16**, 45-69. https://doi.org/10.1007/s40558-015-0047-7

[19] Liu, J.S. (2017) 515K Hotel Reviews Data in Europe. https://www.kaggle.com/datasets/jiashenliu/515k-hotel-reviews-data-in-europe

[20] Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computation*, **9**, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[21] Basiri, M.E., Nemati, S., Abdar, M., Cambria, E. and Acharya, U.R. (2021) ABCDM: An Attention-Based Bidirectional CNN-RNN Deep Model for Sentiment Analysis. *Future Generation Computer Systems*, **115**, 279-294. https://doi.org/10.1016/j.future.2020.08.005

[22] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I. (2017) Attention Is All You Need. https://arxiv.org/abs/1706.03762

[23] https://www.analyticsvidhya.com/blog/2021/03/binary-cross-entropy-log-loss-for-b

[24] Harmanpreetsingh (2017) Labelled Hotel Reviews. https://www.kaggle.com/datasets/harmanpreet93/hotelreviews

[25] Vashishtha, S. and Susan, S. (2020) Fuzzy Interpretation of Word Polarity Scores for Unsupervised Sentiment Analysis. 2020 11*th International Conference on Computing, Communication and Networking Technologies* (*ICCCNT*), Kharagpur, 1-3 July 2020, 1-6. https://doi.org/10.1109/ICCCNT49239.2020.9225646