International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 11 (2019) pp. 227-241

Received: 15 July, 2019; Accepted: 28 October, 2019; Published: 24 December, 2019

Deep Learning with Word Embedding Modeling for a Sentiment Analysis of Online Reviews

Wejdan Ibrahim AlSurayyi [1] , Norah Saleh Alghamdi [2] and Ajith Abraham [3]

[1] Princess Nourah bint Abdulrahman University, College of Computer and Information Sciences, Riyadh, Saudi Arabia.
School of Computing, Dublin City University, Ireland
wialsuryyi@pnu.edu.sa

[2] Princess Nourah bint Abdulrahman University,
College of Computer and Information Sciences, Riyadh, Saudi Arabia.
nosalghamdi@pnu.edu.sa

[3]Machine Intelligence Research Labs (MIR Labs),
Scientific Network for Innovation and Research Excellence, P.O. Box 2259, WA, USA
ajith.abraham@ieee.org

Abstract: **Recently, online buyers have been able to express their opinions about a variety of products, restaurants and services by writing online reviews. Opinions have subsequently become a new, important, and influential source of information for decision-making. This paper implements binary and multiclass sentiment classifications on a dataset of online reviews. The experiments are performed using restaurant reviews from Yelp to predict ratings based on the content of the reviews. This paper investigates various structures of neural networks on restaurant reviews, such as recurrent neural networks (RNNs) with long short-term memory (LSTM), RNNs with bidirectional LSTM (Bi-LSTM) and convolutional neural networks (CNNs). The reviews were first converted into vectors during preprocessing using various features: pretrained word2vec and global vector (GloVe) embedding. The efficacy of these text classification techniques was examined and compared. The classification performance was evaluated using different metrics: the accuracy, confusion matrix, recall, specificity, precision, F1 score, receiver-operating characteristic (ROC) curve, and the area under the curve (AUC). The results showed that the RNN model achieved a better accuracy score with Bi-LSTM for both binary and multiple sentiment classification tasks.**

Keywords: Sentiment analysis, Text mining, Star Rating, LSTM, CNN, RNN

I. Introduction

Social media and e-commerce have recently flourished and become everyday forums for information diffusion online. Users are now able to freely express their opinions about a variety of products and services. Choosing a hotel or a restaurant from thousands of options can be an overwhelming task. Subsequently, people commonly rely on online reviews to benefit from the experience of others. Users' opinions have become an important, new and influential source of information for decision-making. User's opinions benefit not only other users but also business owners who can improve their business services and product quality and devise new marketing strategies [1]. Sentiment analysis, which is one of the natural language processing (NLP) technologies, has been successfully used for the analysis of social media and for the extraction of user opinions about products through reviews. NLP began in the 1950s and has gained much attention in recent years. NLP allows computers to read the text, interpret it, identify important parts, measure sentiment and extract knowledge from user-generated content. It is highly related to the interaction between humans and computers [2]. NLP and text mining (also known as text analytics) is a field of computer science, artificial intelligence (AI) and a branch of machine learning that is based on text analysis and predictive analysis [3]. This research purposed NLP and text mining to the reviews written by users and analyze the relationship between the reviews and ratings to predict the star ratings based on the content of the reviews. The research aimed at answering the following research question: How can text analytics be used to predict review ratings? More specifically, how do the word2vec and global vector (GloVe) models perform at extracting words from reviews with a deep learning approach? This paper is organized as follows. The "Related Work" section discusses the existing work in the field. The "Research Methodology" section describes the background of the methods used in this study. The "Implementation and Experiments" section covers the experimental settings and the datasets used. The experimental results and the discussion of the efficacies of the classifiers are presented in the "Results and Discussion" section, followed by the ending remarks along with a future direction in the "Conclusion and Future Work" section.

II. Related Works

Prior research has used machine learning algorithms to classify text [4],[5],[6], which considered sentiment classification as a type of text classification and used supervised machine learning methods such as naive Bayes

(NB) and support vector machines (SVMs). Guixian et al. [4] proposed a Chinese sentiment analysis method based on extended dictionary. They used three datasets: Data set 1, which had 25,000 reviews that were crawled from the Jingdong and eLong websites; Data set 2, which had 50,000 reviews that were crawled from the review data on the Ctrip and Jingdong websites; and Data set 3, which had 6000 labeled hotel reviews provided by the Tan Songbo team. However, they applied the NB classifier and then compared it with the k-nearest neighbors (KNN) and SVM algorithms. In their experiment, the precision, recall and F1 score are used to evaluate the classification results. Therefore, the feasibility of using the NB classifier to identify the text field is suggested. Furthermore, Michela et al. [5] compared the performance of four classifiers with various parameters: a linear SVM, C4.5, a projective adaptive resonance theory (PART) network, and a NB classifier. They applied a 5-fold cross-validation methodology. They evaluated and compared the following metrics: accuracy, precision, recall and F1 score. The most effective classification model was the SVM, and its accuracy was 97.0%. Additionally, Md Shad et al. [6] used aspect category detection and sentiment classification for 5417 reviews in Hindi. The authors collected user-generated reviews from different online sources on 12 domains. Then, they developed supervised approaches for these two tasks. They implemented NB, decision tree (DT) and sequential minimal optimization (SMO) classifiers. For their experiments, they used MEKA, which handles multiple labels. The effective classification models were the SMO and DT classifiers; their accuracy was 91.62%. In recent years, different studies have proven that the use of deep learning techniques (long short-term memory (LSTM), bidirectional LSTM (Bi-LSTM) and convolutional neural networks (CNNs)) gives competitive results with traditional methods such as SVMs and NB classifiers [7] [8] [9]. The results from deep learning techniques are remarkable for text classification and sentiment analysis. Moreover, many studies have applied different deep learning architectures to several languages, such as English [10][11][12], Arabic [13], Chinese [14][4][15][10], Persian [16], Hindi [6][7] and Tamil [17]. These studies have proven the great performance of sentiment analysis with different languages. Sujata and Parteek [7] collected a dataset of Hindi movie reviews from online newspapers and websites (3 classes). They employed the word2vec model to the dataset to learn vector representations of the words. Then, they applied deep learning techniques convolutional neural network (CNN) with different configuration settings. Thereafter, their CNN model was compared with traditional, state-of-the-art ML algorithms such as NB, KNN, maximum entropy (ME) and SVM. Their results show that deep learning techniques outperform traditional ML approaches. Other performance parameters, such as the precision, recall, F1 score, Kappa score, mean absolute error (MAE) and root-mean-square error (RMSE), were used for each of the CNN models.

Long et al.[8] proposed a method called sentiment embedded semantic space (SESS), which captures the connection between the sentiment space and the semantic space. This model is based on the K-means and CNN algorithms. In addition, they developed a sentiment dictionary based on the HowNet dictionary. They proposed a sentiment classifier built on SESS. It consists of two parts: unsupervised learning and supervised learning. They conducted their experiments on 10,662 movie reviews (MRs), where each MR is associated with a binary sentiment polarity label. They used the accuracy to evaluate the classification. In addition, they built the baseline models with word2vec and constructed the sentiment embedding, which has been shown to give competitive results compared with traditional classifiers such as SVM with NB features (NBSVM) and multinomial naive Bayes (MNB). Their results show that significant improvements can be achieved by a CNN classifier built on the SESS. Moreover, Xiaodong et al. [9] studied aspect-based opinion summaries (AOSs) of reviews that need aspect extraction and sentiment classification. Aspect extraction involves either linguistic analysis (i.e., "supervised labeling") or topic modeling (i.e., "unsupervised labeling"). The authors created two datasets and made them publicly available. The first dataset is the Amazon Smartphone Review (ASR) dataset, which contains 12,700 smartphone reviews. Each review is labeled with respect to five predefined aspects. Sentences belonging to at least one aspect are labeled as having a positive or negative sentiment. The second dataset is the Taobao Skirt Review (TSR) dataset. It contains 18,314 labeled reviews sentences and one million unlabeled review sentences. Each labeled review sentence is labeled with respect to six predefined aspects. Additionally, the sentiments for sentences belonging to at least one aspect are labeled. They pretrained the word embeddings using word2vec, which implements the continuous bag-of-words (CBOW) and skip-gram architectures to learn word vector representations. They proposed a model called a cascaded CNN (C-CNN), which is based on the CNN method. This model contains two levels of CNNs. In addition, they used the F1 score and classification accuracy to evaluate the aspect mapping and sentiment classification performance. All experiments were performed using ten-fold cross-validation. Their results showed that a C-CNN with pretrained word embedding outperforms a cascaded SVM with feature engineering.

Previous works performed a series of experiments to explore the effect of architectural components on the performance of models with hyperparameter tuning. Then, design decisions were discussed in terms of the sentiment classification on large databases; these models were compared and the achieved accuracy was reported. Several recent studies of sentiment analysis using deep learning concentrate on learning vectors as features without using feature engineering.

In deep learning, it is common to use bags-of-words "word2vec" representation for text documents. For example, Asad et al. [11] presented a deep learning-based method (called RNSA) to classify a user's reviews. This model applied on three MR datasets with two classes from IMDB. They used word embedding, which is trained by the word2vec model, sentiment based on lexicon and linguistic knowledge features. Then, word embedding and sentiment are fed into an RNSA that employs a recurrent neural network (RNN), which is composed of the LSTM network classifier. The authors used three performance measures: precision, recall, F1 score. Their experimental results proved that their proposed method outperforms the state-of-the-art methods. In 2019, Junhao et al. [14] proposed a model by incorporating a word2vec model and a stacked Bi-LSTM model. They conducted their experiments on a dataset collected from Weibo (one of the most popular Chinese microblogs), in which each comment is associated with a binary sentiment polarity label. Moreover, they also evaluated the performance of two typical word2vec models: the CBOW and skip-gram models. They used the

accuracy to evaluate the classification. In addition, they also compared their results with other baseline models, such as SVM, logistic regression (LR), CNN, stacked CNN, LSTM and Bi-LSTM models. Their proposed stacked Bi-LSTM model with either CBOW or skip-gram had a better predication accuracy than the other models.

Ref. Paper	No. of Reviews/ Dataset	Sentiment Analysis Tools (NLP)	Models (Machine learning + deep learning)	Best Performance
[11]	50,000/movie reviews (MRs) (IMDB)	▪ word embedding (word2vec) ▪ sentiment (lexicon) ▪ linguistic knowledge	▪ various RNSA methods	RNSA Full F1 score 74% Recall 65% Precision 86%
[7]	7354 Hindi MRs (web crawler)	▪ word embedding (word2vec)	▪ various CNN models ▪ NB ▪ KNN ▪ ME ▪ SVM	95%
[14]	65,536 Chinese comments/web crawler (Weibo)	▪ word2vec models (CBOW and skip-gram)	▪ SVM ▪ LR ▪ CNN ▪ Stacked CNN ▪ LSTM ▪ Bi-LSTM ▪ Stacked Bi-LSTM	Stacked Bi-LSTM 90.3% (skip-gram) 89.5% (CBOW)
[10]	IMDB, Yelp2013, MR, NB4000 and Book4000	▪ sentiment lexicon	RAE ▪ LSTM ▪ Bi-LSTM ▪ CNN ▪ Tree-LSTM ▪ LE-LSTM ▪ ALE-LSTM ▪ WALE-LSTM	WALE-LSTM 89.5% (IMDB) 60.6% (Yelp)
[4]	25,000/Jingdong-eLong + 50,000/Ctrip-Jingdong + 6000/Tan Songbo	▪ extended sentiment dictionary (basic-some field words-polysemic) sentiment	▪ NB ▪ KNN ▪ SVM	NB 86%
[15]	15,000/Ctrip	▪ word2vec models (CBOW and skip-gram) ▪ TF-IDF ▪ word vectors	▪ Bi-LSTM ▪ RNN ▪ CNN ▪ LSTM ▪ NB	Bi-LSTM 92.18%
[12]	SemEval and SST	word embedding (word2vec + GloVe) sentiment Embeddings (HyRank+M-TSWE+SWV-H) refined Embeddings (Re'word2vec + GloVe+ HyRank')	▪ CNN ▪ DAN ▪ Bi-LSTM ▪ Tree-LSTM	Tree-LSTM 90.3% (Binary) 54% (Multiclass)
[5]	726,327 bookings + 353,167 TripAdvisor reviews	▪ unsupervised/(STWV) ▪ supervised/(AS)	linear SVM C4.5 PART ▪ NB	SVM 97.0%
[8]	10,662 MRs	▪ word2vec ▪ HowNet dictionary and sentiment embedding	NBSVM MNB CNN	CNN-SESS 83%

Ref. Paper	No. of Reviews/ Dataset	Sentiment Analysis Tools (NLP)	Models (Machine learning + deep learning)	Best Performance
[19]	75,933 TripAdvisor reviews	▪ sequential labeling with IOB ▪ LDA	Bi-LSTM-CRF	69.60%
[18]	AGNews Sogou Yelp Yelp Binary Yahoo Amazon Amazon Binary	▪ word embedding (word2vec)	CNN LSTM Att-Bi-LSTM CRAN	65.66%
[6]	5,417 online crawls	▪ MEKA	NB DT SMO	SMO/DT 91.62%
[9]	12,700 Amazon Smartphone Review + 18,314 Taobao Skirt Review	▪ IDF ▪ Bigram ▪ word2vec	SVM CNN	CNN 84.87%

Table 1. Summary of the datasets and feature extraction methods used in the literature.

Guixian et al. [15] proposed an improved word representation method that integrates the contribution of sentiment information into the traditional term frequency-inverse document frequency (TF-IDF) algorithm and generates weighted word vectors. The authors used 15,000 hotel comments texts crawled from Ctrip. To obtain distributed representations of words, they used word2vec technology, including the CBOW and skip-gram models. Next, they fed their results into a Bi-LSTM model. Then, they compared their proposed sentiment analysis method with the RNN, CNN, LSTM, and NB sentiment analysis methods. Their experimental results showed that their proposed sentiment analysis method has higher precision, recall, and F1 scores. The method for classifying comments was proven effective with high accuracy. Liang-Chih et al. [12] proposed a word vector refinement model to refine existing pretrained word vectors that can be applied to any pretrained word embedding. Their experimental results showed that their proposed refinement model can improve both conventional word embeddings and their proposed sentiment embeddings for binary, ternary, and fine-grained sentiment classification on the Semantic Evaluation (SemEval) and Stanford Sentiment Treebank (SST) databases. Long et al. [18] proposed a hybrid CNN-RNN attention-based neural network called the convolutional recurrent attention network (CRAN). This model was applied to different publicly available datasets with different classes (AGNews-4 class, Sogou-5 class, Yelp-5 class, Yelp Binary-2 class, DBPedia-14 class, Amazon Full-5 class, Amazon Binary-2 class and Yahoo-10 class). The authors adopted pretrained word embeddings by training an unsupervised word2vec model on the datasets. Finally, they compared their model with several baseline methods.

Many researchers have used approaches involving recursive neural network models for sentiment analyses of online text. For example, Xianghua et al. [10] proposed a lexicon-enhanced LSTM model called LE-LSTM to introduce a sentiment lexicon into LSTM. They also proposed a method to calculate the attention vector in a general sentiment analysis without a target and took two special circumstances as examples: lexicon-enhanced LSTM with attention (ALE-LSTM) and WALE-LSTM. They used five datasets (IMDB, Yelp2013, MR, NB4000 and Book4000). The first three datasets are English datasets, and the last two datasets are Chinese datasets. All the datasets have 2 classes except the Yelp2013 dataset, which has 5 classes. In addition, the authors compared the results of their experiments with the main methods used in sentiment classification, such as the recursive autoencoder (RAE) model, the standard LSTM model, the Bi-LSTM model, the CNN model and the Tree-LSTM model. In 2019, Thang et al. [19] applied a sentiment analysis on a hotel review dataset from TripAdvisor. They implemented topic modeling using latent Dirichlet allocation (LDA) on their dataset to discover the keywords representing each topic. Then, the results were fed into the Bi-LSTM with a conditional random field (Bi-LSTM-CRF) model. However, the authors modified the input and output of the model by combining aspect terms and polarities using sequential labeling with a new inside-outside-beginning (IOB) encoding format. Their aim was to improve services in the hotel industry by identifying the aspect terms with polarities in the reviews. Table 1 below provides a summary of the datasets and various methods in the literature. From the reviewed literature, various deep learning algorithms are designed for sentence classification; however, the use of the Yelp dataset to compare different deep learning algorithms with two-class or multiclass algorithms and the use several pretrained word embeddings for text review classification has not been adequately research. Therefore, this study was designed to close this research gap by comparing the performance of deep learning algorithms with binary and multiclass algorithms using the Yelp dataset with the same parameters as previous works but with different pretrained word embeddings. In addition, the updated version of the Yelp dataset (issued in 2018) has not been used in previous studies. This version has a larger sample size and more reviews, businesses and users than the previous versions. In addition, we use different evaluation metrics, including the accuracy, confusion matrix, recall, specificity, precision, F1 score, receiver-operating characteristic (ROC) curve, and the area under the curve

(AUC), to evaluate the performance of the classifiers. The confusion matrix, recall, specificity, precision, F1 score and ROC curve metrics have not been used before in the literature with the Yelp dataset. Additionally, special hyperparameters have been selected in this study to reduce the training and utilization times to achieve a higher accuracy (more details are given in the "Implementation and Experiments" section).

Table 2. *Summary of the datasets and feature extraction methods used in the literature.*

III. Research Methodology

This section provides an overview of the methodology used in this paper, which can be summarized as follows: data selection and collection, preparation of the data by filtering, checking for missing values, preprocessing the text, tokenizing and creating sequences, applying text representation models (word embedding), and analyzing the data to generate alternate classifications of reviews (e.g., LSTM networks, Bi-LSTM networks and CNNs). Finally, the evaluation of the algorithms and the performance comparison are carried out. To accomplish the objectives of our research, we employ the methodology shown in Figure 1.

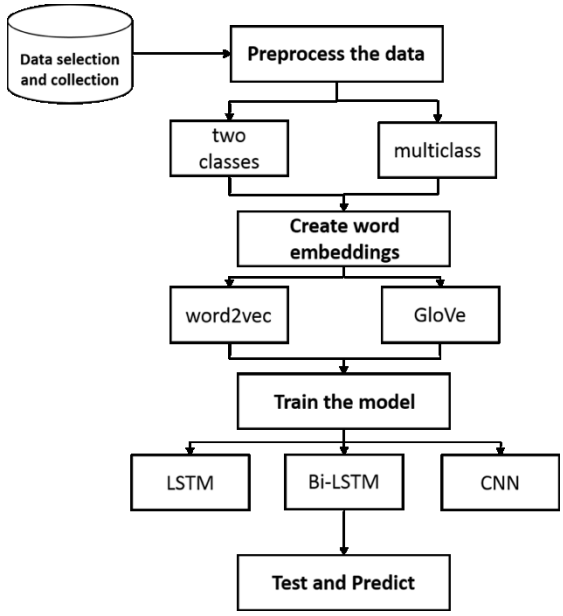

Figure 1. Overall methodology

A. Text Classification

Text classification is also called text categorization or text tagging. It is one of the essential tasks in NLP. It is the process of assigning tags or categories to text according to the content. It describes a general class of problems, for example, predicting the sentiment of tweets, restaurants and MRs as positive or negative, as well as classifying emails as spam or not spam. A classifier takes the text as input and analyzes its content before automatically assigning the relevant categories [3] (see Figure 2).

Figure 2. How does text classification work?

The text is a type of unstructured data and may not be clean. It can be difficult and time consuming to analyze, understand, organize and sort through the textual data using text classifiers, and companies can save time when analyzing textual data, which can help inform business decisions and automate business processes. Some of the reasons why text classification is important include scalability, or easily analyzing millions of texts at a fractional cost, and consistency of criteria, as text classification applies the same criteria to all the data, minimizing the errors when human annotators make mistakes due to inconsistent criteria. Examples of text classification tasks are classifying short texts (e.g., tweets or headlines) and organizing larger sets of documents (e.g., customer reviews). Some of the most famous examples of text classification are sentiment analysis, topic labeling and language detection. Text classifications can be performed in two ways: manually or automatically. Manual annotation is time consuming and expensive, while automatically classifying text is faster and more cost effective. The first step towards training a classifier is feature extraction, which is used to convert each text block into a numerical representation in the form of a vector. Then, a machine learning or deep learning algorithm is fed with the training data consisting of pairs of feature set vectors for each example of text and tags to produce a classification model (see Figure 3).

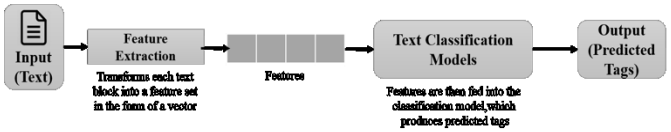

Figure 3. How does text classification work? (in detail)

B. Deep Learning Models

Neural networks are used in deep learning and consist of different interconnected layers ("input layer, hidden layers and output layer") and have similar structures and functions as those in the human brain. They learn from vast amounts of data and use complex algorithms for training. The two kinds of popular neural network models used in this paper are described in this section.

1) Recurrent Neural Networks (RNNs)

An RNN is a model of neural sequences that achieves state-of-the-art performance on important tasks. It can handle sequential data. It considers the current input and the previously received inputs, and it can memorize previous inputs due to internal memory. It can process sequences of inputs using their "memory" (internal state) [20]. RNNs have loops. A loop enables the transmission of information from one step of the network to the next. In the diagram below, a part of a neural network, A, looks at some input xt and outputs a value ht [21] (Figure 4).

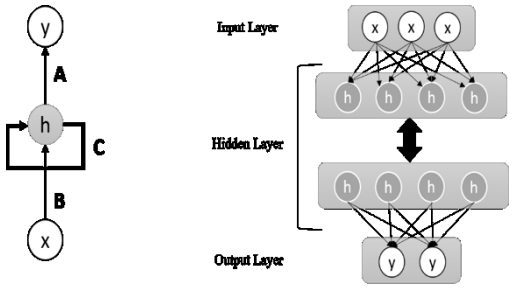

Figure 4. RNN structure and layers

This loop structure enables the input sequence to be available to the neural network. This concept can be better understood by examining the unrolled version (Figure 5) [22].

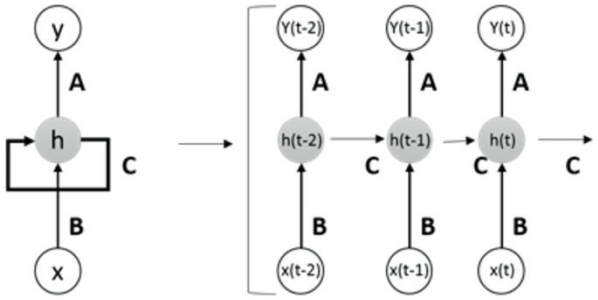

Figure 5. RNN loop structure (unrolled version)

- Long Short-Term Memory (LSTM) Networks are a complex deep learning approach. Hochreiter and Schmidhuber (1997) [22] introduced these networks, and many other researchers refined and popularized them. These networks work very well on a wide range of issues and are now widely used. LSTM networks are a type of RNN capable of learning long-term dependence in sequence prediction problems [21]. Their default behavior is to remember information for long periods of time. LSTM networks involve a three-step process. Step 1 is to forget irrelevant parts of the previous state; step 2 is to selectively update the cell state values; step 3 is to output certain parts of the cell state [22].
- Bidirectional Long Short-Term Memory (Bi-LSTM) Networks have the same architecture as a regular LSTM network but include an additional layer, which means that the signal propagates from past (backward) and future (forward) states at the same time (e.g., from the end of the text to the start of the text). Both LSTM and Bi-LSTM networks are particularly useful in fine-grained sentiment tasks [23].

2) Convolutional Neural Networks (CNNs)

Yann LeCun pioneered CNNs as the director of Facebook's AI research group [24]. In 1988, he built LeNet: the first CNN. It has been used in such tasks as reading ZIP codes and digits. CNNs are a form of feed-forward neural networks, which are generally used to recognize images and classify objects. CNNs are also referred to as "ConvNets". RNNs work by saving a layer's output and feeding it back to the input to predict the layer's output. CNNs consider only the current input, while RNNs consider the current input and the previously received inputs. It can memorize previous inputs due to its internal memory. RNNs can handle sequential data while CNNs cannot. CNNs have four layers: convolution, rectified linear unit (ReLU), pooling and fully connected layers [25]. Every layer has its own functionality and performs feature extraction and discovers hidden patterns. There are some filters in the convolutional layer that perform convolution operations. After extracting feature maps, the next step is to move them to a ReLU layer. The corrected feature map now passes through a pooling layer. Pooling is a downsampling operation that reduces the feature map's dimensionality. Figure 6 shows how CNNs work in practice.

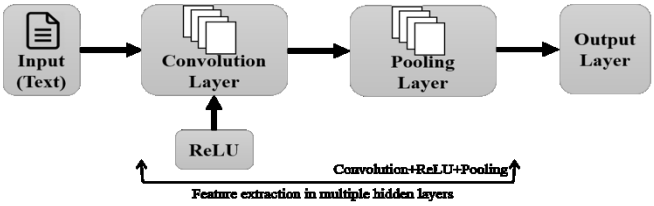

Figure 6. CNN structure and layers

V. Implementation and Experiments

In this section, the implementation strategy and experimental results are presented and analyzed. In addition, the data and the tools used in this paper are explained. Then, the evaluation measures used for classification during the experiments are presented. The Yelp academic dataset is used in the experiments. This dataset is publicly available dataset from Yelp [26] and Kaggle [27]. Yelp is an American corporation located in San Francisco with an online website (yelp.com) and a mobile app that publish crowd-sourced reviews about different businesses [21]. Yelp released a dataset as a part of the Yelp Dataset Challenge, which is a contest for students to conduct research on their 2004-2018 data. This research uses reviews from Yelp.com to ensure high credibility of user opinions posted on Yelp. Yelp uses a filtering algorithm to filter suspicious reviews and minimizes the risk of them appearing on the businesses' pages [28]. Yelp, however, does not delete these filtered reviews but puts them in a list. These datasets ("YelpZip") [29] are available to the public. [30] [1].

Yelp allows users to connect with many businesses from a variety of categories, such as restaurants, cafes, medical clinics, pharmacies, hotels, in four different countries: Canada, USA and some parts of Germany and the UK. Yelp provides users with a way to interact with businesses they visit by rating and reviewing them. Users can also give a star rating from 1 to 5 for a business and can write a text review that clarifies the rating. These ratings are very useful for users who are exploring local business by helping them judge which ones would be the best for them [2]. The Yelp dataset is composed of six compressed JSON files [24]. These files were first converted to CSV files before loading them into Python. The dataset contains 174,567 businesses and 1,326,100 users with 5.26 million reviews and 1,098,324 tips. Three specific data files (business, review and user) will be used in this paper. Table 2 below provides a summary of the used dataset files. This research considers only businesses that are categorized as restaurants. Restaurants are the top category. The bar plot in Figure 7 shows the top categories in the dataset. In total, 2,353,827 reviews (records) were available after filtering.

We used the Yelp reviews dataset, which contains written reviews of restaurants. It has two fields: stars and text, where the stars are the customer's rating from 1 to 5 and the text is the customer's written review. Therefore, it is important to understand the distributions of star ratings by the number of reviews. Figure 8 shows the distribution of star ratings vs the number of reviews. Notice that there are many 4-star and 5-star reviews.

	Records	Features	Data features
business	174,567	13	location data, attributes, review_count, stars and categories, etc. 59,106 different types/categories of businesses (the most popular category is restaurants)
review	5.26 million	9	full review text data including the ID of the user (user_id) who wrote the review and the business_id the review is written for and the stars given
user	1,326,100	22	user_id, name, review count, friends, average number of stars, etc.

Table 2. Files from the Yelp dataset used in this study.

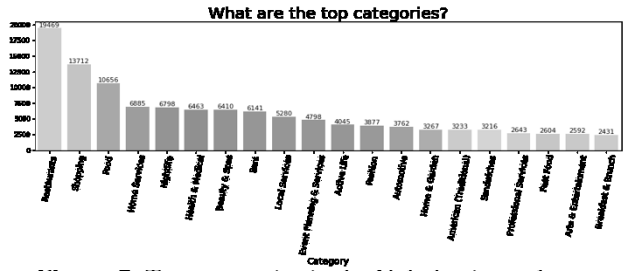

Figure 7. Top categories in the Yelp business dataset

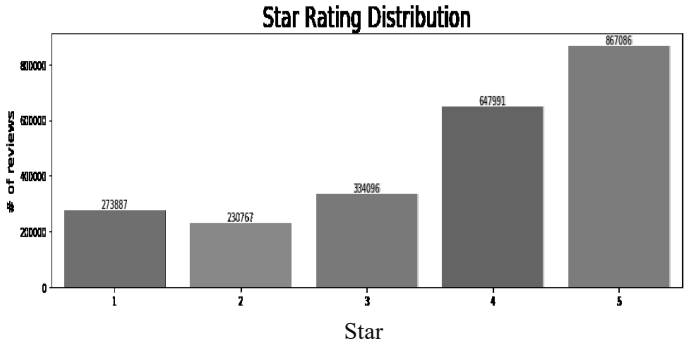

Figure 8. Star ratings vs the number of reviews in the Yelp dataset

Text preprocessing steps are needed to transform the text from human language to a machine-readable format for further processing. Online reviews contain much noise-such as hyperlinks, HTML tags, and informal words, and many words do not have any significant impact on the sentiment of the review. Therefore, the "**NLTK**" library was used to remove such unnecessary text. The text preprocessing steps are listed below.

- Removing punctuation, accent marks and other diacritics;
- Converting words to lower case. It is useless to have the same words in different cases (e.g., "good" and "GOOD");
- Removing hyperlinks;
- Removing stop words;
- Removing white space and any unwanted spaces between words;
- Converting informal words such as "I'll" and "I've" to their formal forms ("I will" and "I have", respectively);
- Using some regular expressions (regexs) to clean the text (e.g., "char filtering");
- Applying tokenization [1]. The Keras tokenizer function is used to divide a sentence into a list of words;
- Creating sequences. Text data must be encoded as numbers to be used as input in deep learning models. The "texts_to_sequences" function was used to make sequences of words by "converting [the] text into [a] numerical representation". A maximum sequence length of 50 was used, which means that a review could have a maximum of 50 words.

Several pretrained models of word embeddings built for obtaining vector representations of words were used. In addition, word2vec and GloVe embeddings were used for comparison. For the pretrained word2vec word embeddings, we created our own word embedding with Gensim library[2]. The parameters used to train this model were the following: the number of embedding dimensions was set to the default value of 100; the window parameter, which is the maximum distance between a target word and the words around the target word, was set to the default value of five; the minimum number of words was four, which is the number of threads used during training. After the model was trained, it was accessible via the word vector (wv) attribute. The converted file was in ASCII format, not binary, so we set binary=False when loading the file. The learned vocabulary included 377,256 tokens (words) [31].

For the pretrained GloVe word embeddings, the smallest GloVe model was used. An 822 MB zip file was downloaded from the GloVe website with four different models (50-, 100-, 200- and 300-dimensional vectors) trained on data from Wikipedia with 6 billion tokens and a 400,000-word vocabulary. In this experiment, we used 100-dimensional vectors [32] [33]. After the entire GloVe or word2vec word embedding file was loaded, we then prepared the embedding layer for the neural network model. The embedding layer is the first hidden layer of a network defined and is supported by the Keras library. Embedding layers require integer-encoded input data; thus, each word was represented by a unique integer. In this experiment, the embedding layer has a vocabulary of 20,000 words (e.g., integer-encoded words from 0 to 19,999, inclusive), an input length of 50, and a vector space of 100 dimensions into which words will be embedded. To test the effectiveness of our model, we used two different techniques. First, the Yelp datasets were transformed into binary (i.e., two-class) data: positive and negative reviews.

[1] Tokenization is one of the essential parts of NLP. A vocabulary size of 20,000 is used, which represents the maximum number of unique words that are used.

[2] https://radimrehurek.com/gensim/index.html

Negative labels were assigned to ratings of 2 stars and below. Positive labels were assigned to ratings of 4 stars, and above, and neutral, 3 star reviews were excluded. Second, the Yelp dataset was adapted to the task of multiclass sentiment analysis, where the data have five levels from 1 to 5; a higher value is better. The details of our datasets are shown in Table 3. All datasets were randomly divided into training data, validation data and test data at a 3:1:1 ratio.

Name	#Train	#Dev	#Test	Classes	Vocabulary size
Yelp	1,412,295	470,766	470,766	5	20,000
Yelp Binary	1,211,838	403,946	403,947	2	20,000

Table 3. Experimental data.

In this experiment, LSTM was implemented with the 100-dimensional embedding layer; these vectors then pass to the LSTM layer. The efficient adaptive moment estimation (Adam) gradient descent optimization algorithm was used to improve the model, and the accuracy was calculated at the end of each batch. The model was trained for 20 epochs or 20 passes through the training data with a batch size of 128. The Bi-LSTM was also implemented. It has the same architecture and parameters as the LSTM. The hyperparameter values for LSTM and Bi-LSTM are shown in Table 4.

Parameter	Value	Parameter	Value
Input length	50	Embedding size	100
LSTM size	100	Hidden layer size	128
Dropout	0.25	Recurrent dropout	0.25
Activation	Sigmoid/ softmax	Optimizer	Adam
Cross-entropy loss	Binary/ categorical	Epochs	20
Batch size	128	Output	Sigmoid

Table 4. LSTM/Bi-LSTM hyperparameter values.

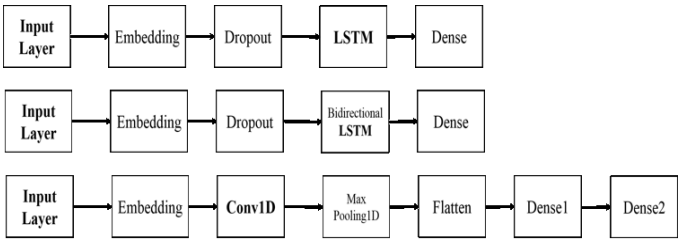

Figure 9. Plot of the defined LSTM, Bi-LSTM and CNN classification models

The third model implemented was a CNN model. First, the length of the input sequences was defined in the input layer by the embedding layer. Then a 1D convolutional (Conv1D) layer was used with 32 filters and a kernel size of 8 with a ReLU activation function that was set to the number of words to read at once. This layer was followed by 1D max pooling (MaxPooling1D) layer with size of two, which was used to consolidate the output from the convolutional layer. Next, a flattened layer was used to reduce the dimensions of the output. Then, a dense layer was applied. The output layer, epochs, batch size and cross-entropy loss used the same values as those used in LSTM during training. Figure 9 shows a plot of the defined LSTM, Bi-LSTM and CNN classification models.

VI. Results and Discussion

In this section, the results of various models with different word embeddings in two or more classification problems are shown. The accuracy and the AUC are the metrics that were used in this work. The accuracy can be obtained by using the following formula:

$$Accuracy = \frac{\#\ correctly\ classified\ items}{\#\ all\ classified\ items} \quad (1)$$

The AUC (referring to area under the ROC curve) can be used to compare the performance of two or more classifiers. It gives a measure of the relative share of true positive and false positive rates (TPR and FPR, respectively) depending on a threshold [34]. The ROC curve is a visual way to evaluate the performance of classifiers. It is made by plotting the TPR, or recall, against the FPR [35] (Figure 10). The following formulas are used:

$$TPR = \frac{TP}{TP + FN} \quad (2)$$

$$FPR = \frac{FP}{FP + TN} \quad (3)$$

In Table 5, the achieved results of the six models along with pretrained GloVe and word2vec word embeddings are illustrated. The best results for each model are highlighted in bold. For both binary and multiclass classification problems, the best models are the LSTM and Bi-LSTM models. The best overall model is the Bi-LSTM model with GloVe embeddings, performing the best across the two-class dataset with an accuracy of 95.75% and an AUC score of 0.989. The worst model is the CNN model (both classifiers). An interesting result is the difference in the performance between the CNN models using GloVe and word2vec embeddings, which are shown in Figure 10 and 11.

Table 5. Model results for the Yelp reviews dataset.

	Model	Score Using split_train_test Word Embedding			
		GloVe		word2vec	
		Accuracy (%)	AUC	Accuracy (%)	AUC
	Binary Classification				
1	LSTM	95.655	**0.989**	95.666	**0.989**
2	Bi-LSTM	**95.759**	**0.989**	95.705	**0.989**
3	CNN	93.195	0.975	94.294	**0.989**
	Multiclass Classification				
4	LSTM	63.666	**0.894**	63.454	**0.894**
5	Bi-LSTM	**64.028**	**0.894**	**63.718**	**0.894**
6	CNN	58.629	0.652	60.452	0.872

Referring to Figures 10 and 11, our model was able to produce an AUC of 0.989, where a 1.0 is the highest possible AUC score. The worst AUC score was approximately 0.65, but some examples above achieved an AUC score of 0.98. Despite the variability in these results, the lower limit of our AUC

score is still very high, which shows that our model performs well. To achieve this result, we trained the neural network on the best set of fine-tuned parameters.

Another evaluation measure used to rank the performance of a classification algorithm is a confusion matrix (Table 6).

		Actual	
		Positive	**Negative**
Predicted	**Positive**	True positives (TPs)	False negatives (FNs)
	Negative	False positives (FPs)	True negatives (TNs)

Table 6. Confusion Matrix.

In Figure 12 and Figure 13, the confusion matrix for the six approaches is shown. We observed that both the LSTM and Bi-LSTM models outperform the CNN model.

On the other hand, the confusion matrix can compare other performance measures such as the precision, specificity, recall, and F1 score. These measures help to provide more detail about the model. The precision is used to indicate the correctly predicted positive class from the total predicted patterns. Moreover, the specificity measures the proportion of true negatives (TNs) that are correctly identified. The recall, on the other hand, indicates the correctly classified positive pattern. The F1 score is used to indicate the weighted harmonic mean of the precision and recall [36].

The precision, specificity, recall, and F1 score can be obtained by using the following formulas:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (4)$$

$$\text{Specificity} = \frac{TN}{TN + FP} \quad (5)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (6)$$

$$\text{F1score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (7)$$

Moreover, the evaluation measurements for all six approaches were compared (Table 7). We observe that the evaluation measurements for Bi-LSTM with GloVe embeddings are the best out of both classifiers

Generally, it is difficult to directly compare the results of different studies since there are often differences in the architecture, the parameter values, the partitioning and preprocessing the training and testing datasets, which is why we tried different combinations of word embeddings and text classifiers on binary and multiclass datasets in our research so that we can compare their performance collectively and accurately. To compare our results with previous studies that used the same datasets as our study, we observed the following findings. For multiple classifications with the LSTM model, Bo and Binwen [37] achieved an accuracy of 59.91%, while we achieved a higher accuracy of 63.66%. Manideep Bollu examined binary and multiple classifications by using LSTM and Bi-LSTM. He achieved an accuracy of 54.9% for multiple classifications with the LSTM model, while we gained a higher accuracy of 63.66%; additionally, he achieved an accuracy of 91.2% for binary classifications, while we achieved a 95.6% accuracy. In addition, Bi-LSTM yielded a higher accuracy for multiple classifications (64.02% vs 58.6% in our study and Manideep's study, respectively) and for binary classifications (95.7% vs. 94.4% in our study and Manideep's study, respectively) [21]. Our model outperformed the models from previous studies due to the parameters' values and word embeddings, which help to achieve the best performance compared with other studies.

using split_train_test (%)					
Model	**A**	**P**	**S**	**R**	**F1-S**
Two-class					
LSTM-GloVe	95.655	95.66	95.66	95.65	95.66
LSTM-word2vec	95.666	95.66	95.65	95.67	95.66
Bi-LSTM-GloVe	**95.759**	**95.75**	**95.76**	**95.76**	**95.76**
Bi-LSTM-word2vec	95.705	95.70	95.750	95.71	95.70
CNN-GloVe	93.195	93.10	92.94	93.19	93.10
CNN-word2vec	94.294	94.25	94.400	94.29	94.26
Multiclass					
LSTM-GloVe	63.666	62.67	62.60	63.67	63.02
LSTM-word2vec	63.454	62.39	62.03	63.45	62.71
Bi-LSTM-GloVe	**64.028**	**62.91**	**62.92**	**64.03**	**63.23**
Bi-LSTM-word2vec	63.718	62.66	62.65	63.72	62.97
CNN-GloVe	58.629	59.31	59.11	48.80	47.30
CNN-word2vec	60.452	60.70	60.72	60.45	60.42

Table 7. Evaluation Measurement for The Six Approaches.

VII. Conclusions and Future Work

In this paper, we compared word embedding and neural network-based approaches for sentiment classification. This paper had the goal of discovering which models perform better across different datasets. We conducted experiments on the Yelp dataset with different classification problems, including binary classes and multiple classes. Additionally, incorporating sentiment information into word embeddings during training yielded good results in our datasets. Subsequently, we used LSTM, Bi-LSTM and CNN models using pretrained GloVe embeddings and learned word2vec embeddings. In most cases, pretrained GloVe embeddings were better features than pretrained word2vec embeddings. In this research, we analyzed how word embeddings, feature extraction and the number of classes affect the classification results.

Moreover, the evaluation measures for the classification models were presented. We used a range of different performance measurements to demonstrate the effectiveness of our model: the accuracy, precision, specificity, recall, F1 score, AUC and ROC curve. The results of all our experiments and the comparison of our results with some other published works were discussed. Finally, a better accuracy score was achieved using the RNN model, which showed that Bi-LSTM models performed well and that both LSTM and Bi-LSTM models are particularly good at both binary and multiple sentiment tasks.

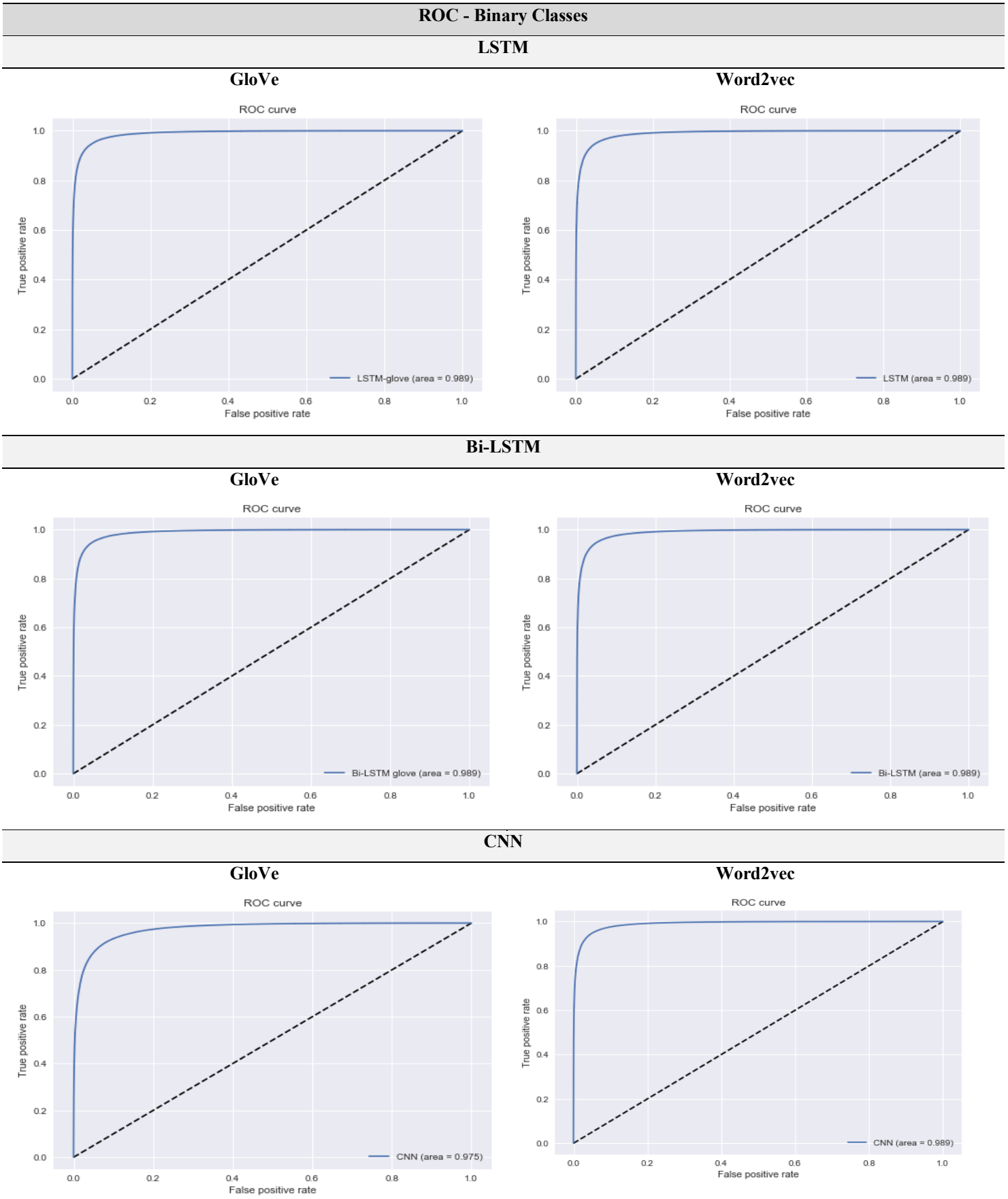

Figure 10. ROC of the CNN, LSTM, and Bi-LSTM models (binary classification)

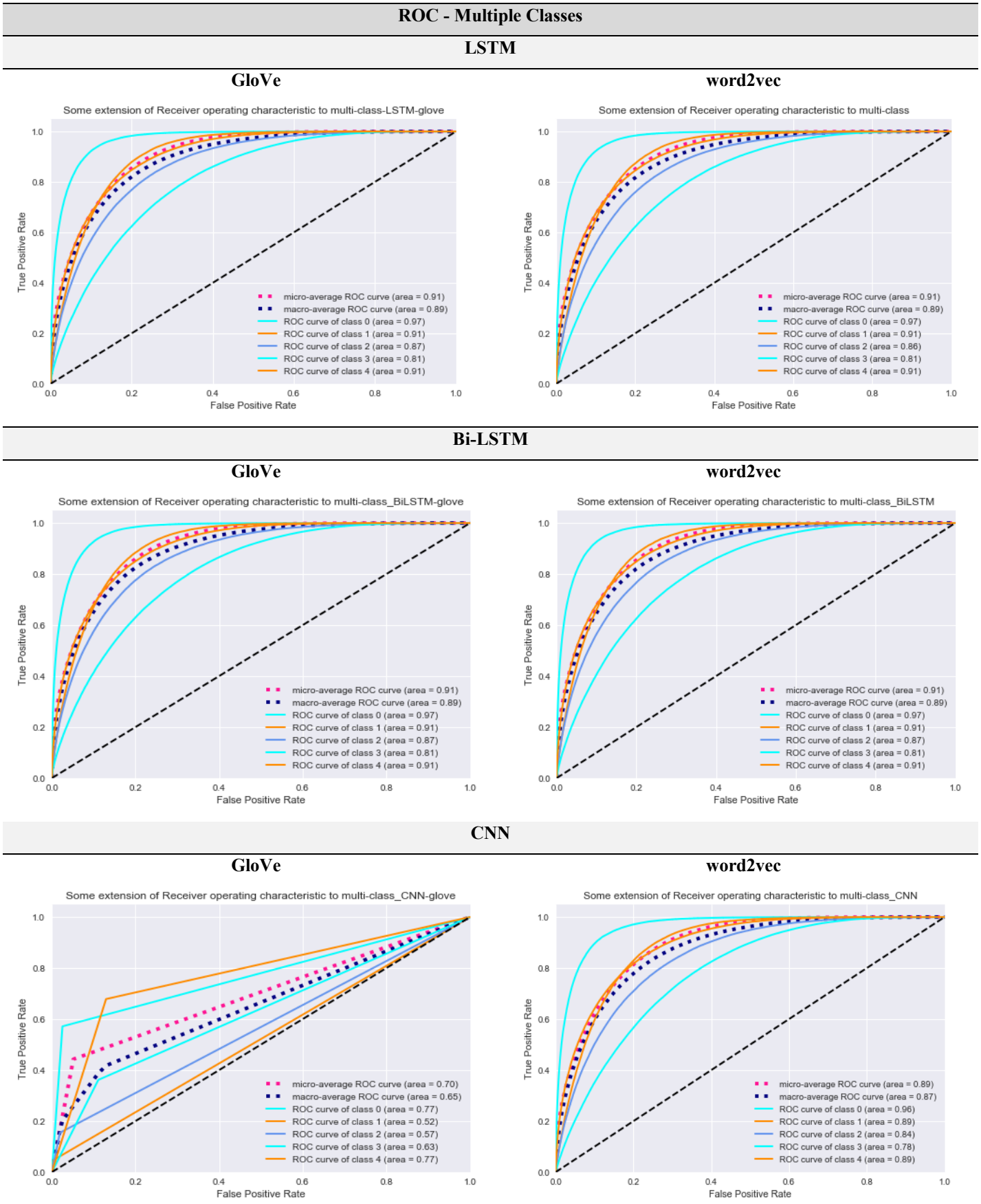

Figure 11. ROC of the CNN, LSTM, and Bi-LSTM model (multiclass classification)

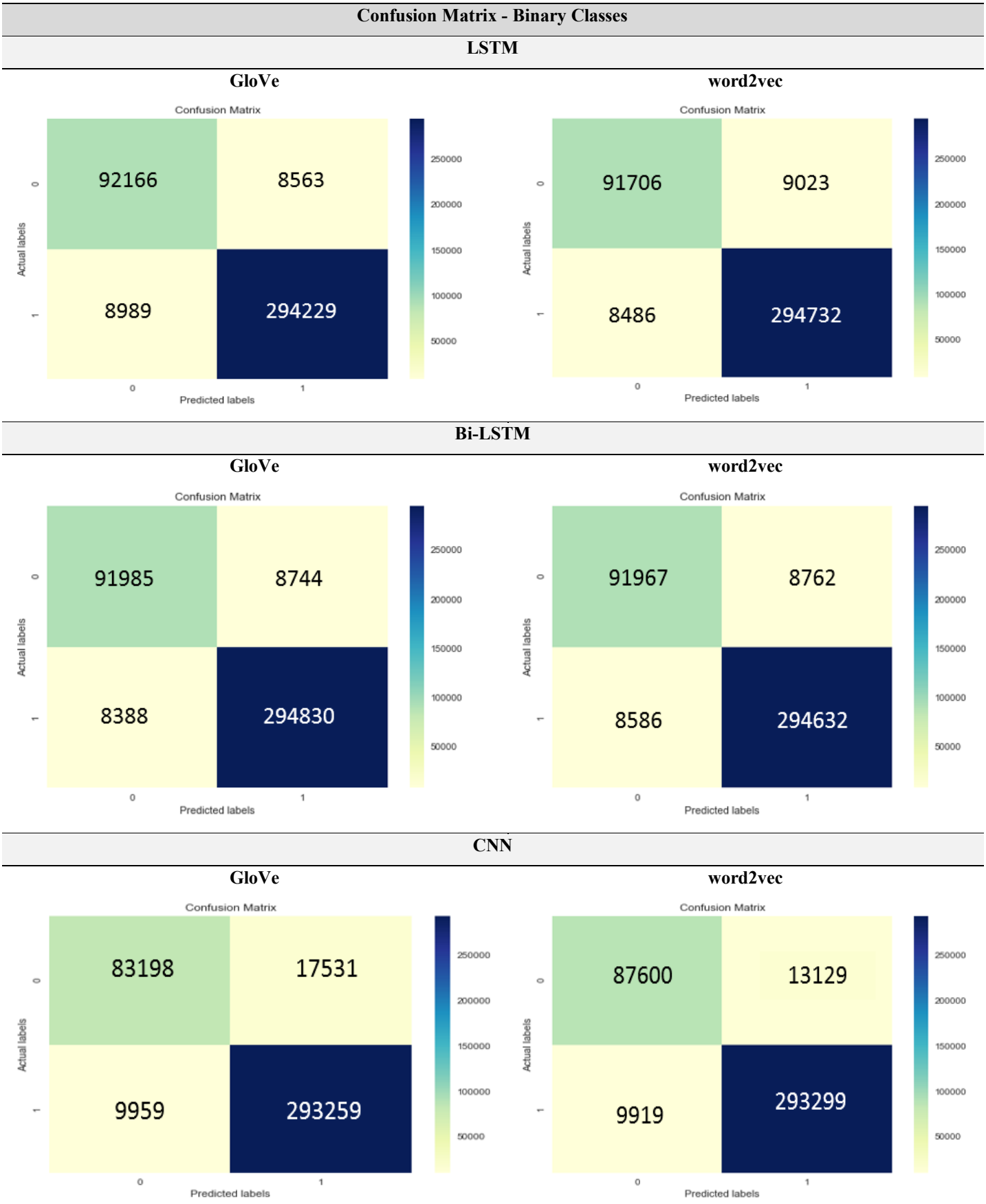

Figure 12. Confusion matrices of the CNN, LSTM, and Bi-LSTM (Two classes)

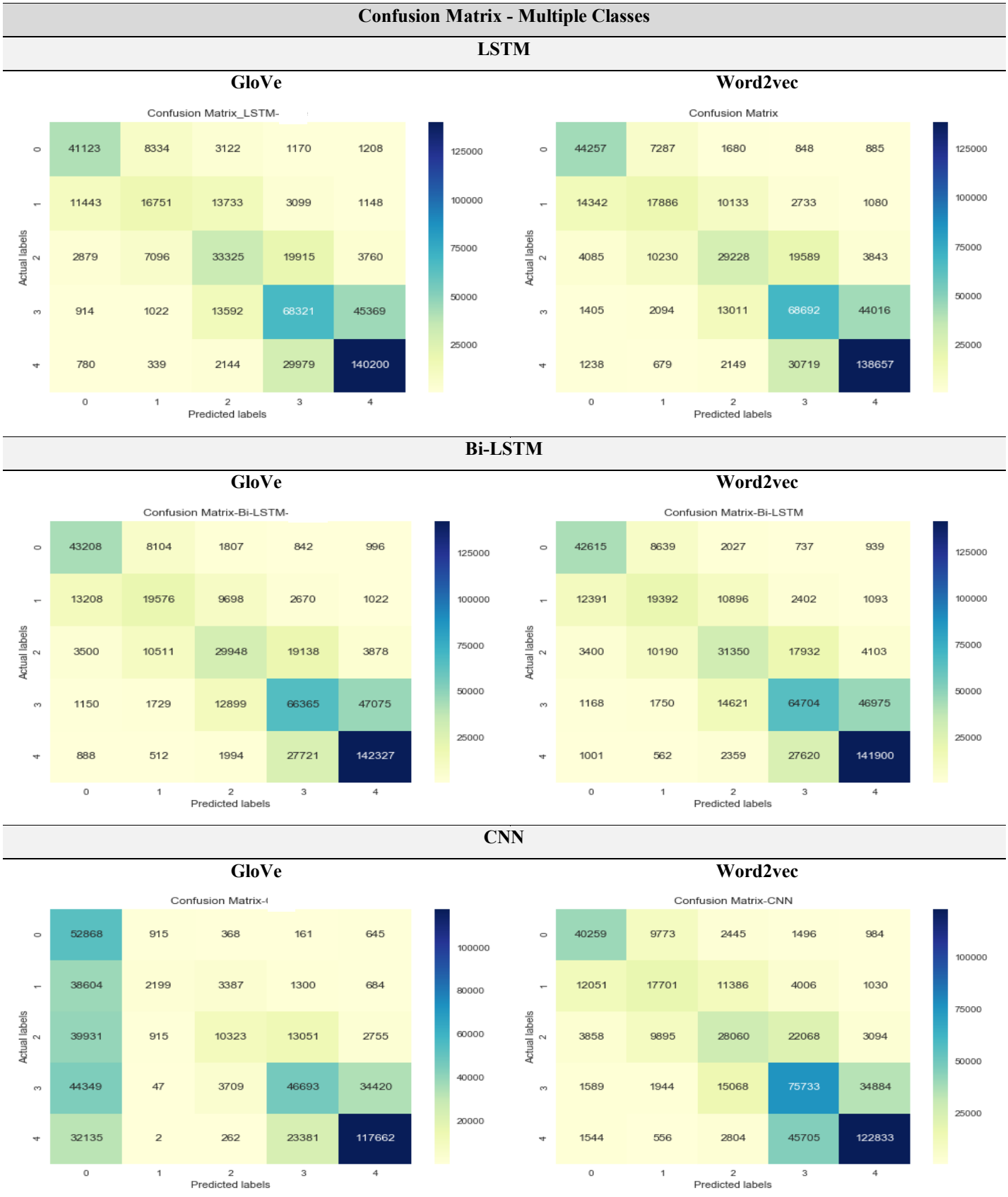

Figure 13. Confusion matrices of the CNN, LSTM, and Bi-LSTM models (multiclass classification)

Possible directions for future work include the following:

- Conducting additional experiments using other deep learning methods such as gated recurrent unit (GRU) and multilayer perceptron (MLP) networks for sentiment categorization of user text reviews;
- Using multiple languages, such as Arabic reviews, not just English reviews;
- Trying a large number of epochs when using a graphics processing unit (GPU) allows more training in a smaller time frame, which could allow us to run over 2000 epochs in less than two hours;
- Experimenting using another evaluation metric such as the mean squared error (MSE);
- Using different feature extraction methods, such as bag-of-word term frequency-inverse document frequency (TF-IDF);
- Conducting an empirical study on the effect of hyperparameters on the overall performance in the sentiment classification task, such as the word embedding dimensions, number of hidden units and activation functions;
- Using 10-fold cross-validation to evaluate different hyperparameters for the deep neural methods.

Acknowledgment

This research was funded by the Deanship of Scientific Research at Princess Nourah bint Abdulrahman University through the Fast-track Research Funding Program.

References

[1] N. Li, "Sentiment Features for Yelp Not-recommended Online Reviews Study," *ProQuest Diss. Theses*, p. 69, 2018.

[2] C. Science, "Big Data Analytics Case Study – Yelp Dataset," 2017.

[3] A. S. J Abu Hammad and A. El-Halees, "An Approach for Detecting Spam in Arabic Opinion Reviews A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology," no. January, 2013.

[4] G. Xu, Z. Yu, H. Yao, F. Li, Y. Meng, and X. Wu, "Chinese Text Sentiment Analysis Based on Extended Sentiment Dictionary," *IEEE Access*, vol. 7, pp. 43749–43762, 2019.

[5] M. Fazzolari, V. Cozza, M. Petrocchi, and A. Spognardi, "A Study on Text-Score Disagreement in Online Reviews," *Cognit. Comput.*, vol. 9, no. 5, pp. 689–701, 2017.

[6] M. S. Akhtar, A. Ekbal, and P. Bhattacharyya, "Aspect Based Sentiment Analysis: Category Detection and Sentiment Classification for Hindi," in *International Conference on Intelligent Text Processing and Computational Linguistics*, Springer, Cham, 2018, pp. 246–257.

[7] S. Rani and P. Kumar, "Deep Learning Based Sentiment Analysis Using Convolution Neural Network," *Arab. J. Sci. Eng.*, vol. 44, pp. 3305–3314, 2019.

[8] J. Jiang *et al.*, "Sentiment Embedded Semantic Space for More Accurate Sentiment Analysis," in *International Conference on Knowledge Science, Engineering and Management*, Springer, Cham, 2018, pp. 221–231.

[9] X. Gu, Y. Gu, and H. Wu, "Cascaded Convolutional Neural Networks for Aspect-Based Opinion Summary," *Neural Process. Lett.*, vol. 46, no. 2, pp. 581–594, 2017.

[10] X. Fu, J. Yang, J. Li, M. Fang, and H. Wang, "Lexicon-Enhanced LSTM with Attention for General Sentiment Analysis," *IEEE Access*, vol. 6, pp. 71884–71891, 2018.

[11] A. Abdi, S. M. Shamsuddin, S. Hasan, and J. Piran, "Deep learning-based sentiment classification of evaluative text based on Multi-feature fusion," *Inf. Process. \& Manag.*, vol. 56, pp. 1245--1259, 2019.

[12] L. C. Yu, J. Wang, K. Robert Lai, and X. Zhang, "Refining Word Embeddings Using Intensity Scores for Sentiment Analysis," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 26, no. 3, pp. 671–681, 2018.

[13] R. M. Alahmary, H. Z. Al-Dossari, and A. Z. Emam, "Sentiment analysis of saudi dialect using deep learning techniques," *ICEIC 2019 - Int. Conf. Electron. Information, Commun.*, pp. 1–6, 2019.

[14] J. Zhou, Y. Lu, H. N. Dai, H. Wang, and H. Xiao, "Sentiment analysis of Chinese microblog based on stacked bidirectional LSTM," *IEEE Access*, vol. 7, pp. 38856–38866, 2019.

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

[16] B. Roshanfekr, S. Khadivi, and M. Rahmati, "Sentiment analysis using deep learning on Persian texts," *2017 25th Iran. Conf. Electr. Eng. ICEE 2017*, no. lCEE20 17, pp. 1503–1508, 2017.

[17] R. Padmamala and V. Prema, "Sentiment analysis of online Tamil contents using recursive neural network models approach for Tamil language," *2017 IEEE Int. Conf. Smart Technol. Manag. Comput. Commun. Control. Energy Mater. ICSTM 2017 - Proc.*, no. August, pp. 28–31, 2017.

[18] L. Guo, D. Zhang, L. Wang, H. Wang, and B. Cui, "CRAN: A Hybrid CNN-RNN Attention-Based Model for Text Classification," in *International Conference on Conceptual Modeling*, Springer, Cham, 2018, pp. 571–585.

[19] T. Tran, H. Ba, and V.-N. Huynh, "Measuring Hotel Review Sentiment: An Aspect-Based Sentiment Analysis Approach," in *International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making*, Springer, Cham, 2019, pp. 393–405.

[20] N. Shrestha, "Deep Learning Implementation for Comparison of User Reviews and Ratings," no. May, 2017.

[21] Manideep Bollu, "NEW SOCIAL MEDIA SENTIMENT ANALYSIS ALGORITHM FOR BUSINESSES COMPETITION," vol. 91, pp. 399–404, 2017.

[22] W. Zaremba, I. Sutskever, O. Vinyals, and G. Brain, "RECURRENT NEURAL NETWORK REGULARIZATION," 2015.

[23] J. Barnes, R. Klinger, and S. S. im Walde, "Assessing

State-of-the-Art Sentiment Models on State-of-the-Art Sentiment Datasets," pp. 2–12, 2017.
[24] "Turing Award Won by 3 Pioneers in Artificial Intelligence - The New York Times." [Online]. Available: https://www.nytimes.com/2019/03/27/technology/turing-award-ai.html. [Accessed: 17-May-2019].
[25] A. Salinca, "Convolutional Neural Networks for Sentiment Classification on Business Reviews," 2017.
[26] "Yelp Dataset." [Online]. Available: https://www.yelp.com/dataset. [Accessed: 24-Feb-2019].
[27] "Yelp Dataset | Kaggle." [Online]. Available: https://www.kaggle.com/yelp-dataset/yelp-dataset. [Accessed: 24-Feb-2019].
[28] "Why Yelp Has A Review Filter - Yelp." [Online]. Available: https://www.yelpblog.com/2009/10/why-yelp-has-a-review-filter. [Accessed: 06-Apr-2019].
[29] "YelpZip dataset – ODDS." [Online]. Available: http://odds.cs.stonybrook.edu/yelpzip-dataset/. [Accessed: 06-Apr-2019].
[30] A. Mukherjee, V. Venkataraman, B. Liu, and N. Glance, "Fake Review Detection: Classification and Analysis of Real and Pseudo Reviews."
[31] J. Brownlee, "Deep Learning for Natural Language Processing (Develop Deep Learning Models for Natural Language in Python)," p. 413, 2018.
[32] "GloVe: Global Vectors for Word Representation." [Online]. Available: https://nlp.stanford.edu/projects/glove/. [Accessed: 06-Apr-2019].
[33] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation."
[34] D. J. Hand, "Measuring classifier performance: a coherent alternative to the area under the ROC curve," *Mach Learn*, vol. 77, pp. 103–123, 2009.
[35] S. Agarwal *et al.*, "Generalization Bounds for the Area Under the ROC Curve * Thore Graepel Sariel Har-Peled," 2005.
[36] H. Manning, C., Raghavan, P. and Schütze, "Introduction to information retrieval," 2010.
[37] B. Wang and B. Fan, "Attention-based Hierarchical LSTM Model for Document Sentiment Classification," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 435, p. 012051, 2018.