Polarity Detection of Dialectal Arabic using Deep Learning Models

Saleh M. Mohamed*, Ensaf Hussein Mohamed, Mohamed A. Belal
Department of Computer Science, Helwan University
Cairo, Egypt

***Abstract*—With the evolution of a new era of technology and social media networks, as well as an increase in Arabs sharing their point of view, it became necessary that this research be conducted. Sentiment analysis is concerned with identifying and extracting opinionated phrases from reviews or tweets. Specifically, to determine whether a given tweet is positive, negative, or neutral. Dialectical Arabic poses difficulties for sentiment analysis. In this paper, four deep learning models are presented, to be specific convolution neural networks (CNN), long short-term memory (LSTM), a hybrid of (CNN-LSTM), and Bidirectional LSTMs (BiLSTM), to determine the tweets polarities written in dialectal Arabic. The performance of the four models is validated on the used corpus with the use of word embedding and applying the (k-Fold Cross-Validation) method. The results show that CNN outperforms the others achieving an accuracy of 99.65%.**

***Keywords*—Sentiment analysis; word embedding; sentiment classification; dialectical arabic; deep learning**

I. Introduction

Sentiment analysis is a kind of natural language processing (NLP), where NLP, or computational linguistics, is the scientific research on human language from a computational perspective [1]. Natural language processing [2] is a large-scale field that includes applications and exploration such as translation, generation, understanding of human language, speech & named entity recognition, question answering, and information retrieval, and relationship extraction. Sentiment analysis (SA) uses natural language processing, statistical data, or machine learning techniques to extract the sentiment content of a text. SA, also called opinion analysis in the literature, is a process of automatically identifying opinions on certain topics in a text, whether they are "positive" or "negative" opinions.

Analysis of opinions or feelings continues to attract interest in industry and academics. Nowadays, sentiment analysis is extensively used in various languages. While considerable progress has been made in developing models to analyze sentiment, the field remains an active field of research for many languages throughout the world, especially for Arabic as the fifth most widely used and fourth most frequently used language on the Internet [3]. The analysis of Arabic language sentiments is still limited and considered difficult for a variety of reasons: First, there are very complex structures in the Arabic language. Second, the limited resources for Arabic SA make it difficult to find. Third, it contains a lot of morphological and highly ambiguous terms, many irregular structures, and a wide range of dialectal varieties without writing standards. The complexity of the Arabic language, as discussed earlier, makes it fly in the face of most NLP applications [4].

Arabic is divided into three major categories [5]: 1) classical Arabic (CA), which is the language of the Quran; 2) modern standard Arabic (MSA), a standardized official language that can be written in the news and taught in schools; and 3) dialectal Arabic (DA), used in daily life and oral communication. it is usually an MSA mix of one or more Arab dialects used on social media [6].

Using different dialects on social media, allowing Arabic users to express their thoughts freely, complicates SA. In terms of phonology, morphology, lexical choice, and syntax, Arabic dialects are significantly different from MSA. The dialects of Arabic are divided into [7]:

- Egyptian Arabic (EA): Egyptian and Sudanian Arabic.
- Levantine (LA): Lebanese, Syrian, Palestinian, and Jordanian Arabic.
- Gulf Arabic (GA): Gulf Arabic for the Gulf region.
- Iraqi (IA): Iraqi Arabic.
- Maghrebi (MA): Maghrebi Arabic for Morocco, Algeria, Tunisia, Mauritania, and Libya.

The majority of previous Arabic sentiment analysis research was done on MSA. Where dialects are used, Egyptian (MSA/Egyptian) was the favorite one. Research has been conducted with dialects of (Lebanese, Syrian, Iraqi, Libyan, Algerian, Tunisian, and Sudanese). There are different approaches used in Arabic SA and its dialects. SA approaches are classified into four classes: supervised, unsupervised, semi-supervised, and hybrid [8].

Recently, the world has witnessed a great revolution in deep learning, which has become the cornerstone of many improvements in many fields. The English NLP work began early using deep learning models, and then Arabic NLP. The use of deep learning for Arabic SA has recently received greater attention, showing significant performance improvements [9].

Since efforts to apply deep learning are still limited, further experimental work in this field is needed. This paper focuses on the Arabic language, especially Colloquial Arabic with an Egyptian dialect, and introduces different deep learning models based on word embeddings to automatically detect the polarity of tweets as positive or negative.

*Corresponding Author.

The main contribution of this research is:

Four different deep learning models were proposed using convolution neural networks (CNN), long short-term memory (LSTM), a hybrid of (CNN-LSTM), and bidirectional LSTM (BiLSTM) on the corpus [10].

- The four models are presented with a comparative evaluation using a word embedding technique that represents words into vectors called "continuous bag of words" (CBOW)
- The proposed model outperforms previous related models. It outperforms the model presented by Mohammed and Kora [11], which uses the same models over the same corpus but differs by 24% in CNN, 10% in LSTM.
- The experiments were extended by applying the bidirectional LSTMs (BiLSTM).

The rest of the paper is organized as follows: In Section 2, related work in Arabic SA is presented. The method and materials are presented in Section 3. Section 4 presents the results of the experiments and discussion. Finally, Section 5 discusses the conclusion and our future work.

II. Related Work

The literature presents several works of Arabic sentiment analysis. In this section, the published works of the last six years are covered in this area.

Many papers have been conducted between the years 2005 and 2020 that applied deep learning approaches. 77 papers work on Arabic with its dialects. 18 models of deep learning were applied by the researchers.CNN and RNN are considered the most commonly used models [12]. In the papers sampled in [13], modern standard Arabic (MSA) has been widely used among other types and Egyptian (MSA/Egyptian) is preferred when dialects are used.

There have been a few studies that work on DA or mix both MSA and DA, specifically (MSA/Egyptian). Table I summarises these studies by author, year, and the used methodology. Also, the results of these studies and the used datasets are shown in Table II.

Attia et al. [14] built a multilingual system with multi-class sentiment analysis using CNN. They applied their system to three datasets with three different languages, which are Arabic, English, and German, but here focusing on the Arabic language. For the Arabic language, they used the Arabic Sentiment Tweets Dataset (ASTD) dataset. The ASTD dataset consists of 10.6k tweets. The tweets were gathered from Egypt Trends and were not identified with a specific topic. Their system achieved an accuracy of 67.93% on the ASTD dataset.

Heikal et al. [15] applied two models, which are CNN and LSTM, to the same ASTD dataset. They also used multi-class sentiment analysis. The accuracy of their models is that CNN achieved 64.30% and LSTM achieved 64.75%. Also, similar work by Elnagar et al. [16] applied several models, which are CNN, LSTM, and CNN-LSTM. They tested their models on the Books Reviews in the Arabic Dataset (BRAD). The BRAD dataset consists of 6.9k tweets in different Arabic dialects. CNN achieved an accuracy value of 89.61%, while LSTM and CNN-LSTM achieved 90.05% and 90.02%, respectively.

Another research by Abdellaoui et al. [17] proposed CNN and LSTM on two different datasets, which are ASTD and TEAD. The TEAD dataset consists of 6 million tweets that combine MSA and DA. They applied CNN and LSTM on both datasets.CNN and LSTM achieved precision values of 79% and 81%, respectively, on the ASTD data set, while they achieved 86% and 87.5% on the TEAD dataset.

TABLE I. Studies by Methodology

Ref	Authors	Year	Methodology
[14]	Attia et al.	2016	CNN
[15]	Heikal et al.	2018	CNN LSTM
[16]	Elnagar et al.	2018	CNN LSTM CNN-LSTM
[17]	Abdellaoui et al.	2018	CNN LSTM
[18]	Abdullah et al.	2018	CNN-LSTM
[19]	Abu et al.	2019	LSTM BiLSTM
[20]	L. H. Baniata and S. Park.	2016	CNN-BiLSTM BiLSTM-CNN
[11]	Mohammed and Kora	2019	CNN LSTM CNN-LSTM

TABLE II. Studies by Experimental Results

Ref	DataSet	Results (Accuracy A, F1-Score F, Precision P, Spearman Correlation Scores S) %
[14]	ASTD	CNN (A=67.93, F=29.82)
[15]		CNN (A=64.30, F=64.09) LSTM (A=64.75, F=62.08)
[16]	BRAD	CNN (A=89.61) LSTM (A=90.05) CNN-LSTM (A=90.02)
[17]	ASTD	CNN (P=79) LSTM (P =81)
	TEAD	CNN (P =86) LSTM (P =87.5)
[18]	SemEval 2018 Task 1	CNN-LSTM (S=81.80)
[19]	ASTD	LSTM (A=42.00) BiLSTM (A=41.30)
	LABR	LSTM (A=42.00) BiLSTM (A=41.30)
	ShamiSenti corpora	LSTM (A=64.70) BiLSTM (A=61.80)
[20]	LABR	CNN-BiLSTM (A=86.43) BiLSTM-CNN (A=66.26)
[10]	Corpus on Arabic Egyptian tweets	CNN (A=75.72) LSTM (A=81.31) CNN-LSTM (A=88.05)

Abu et al. [19] developed the Shami-Senti corpus, which is regarded as the first Levantine corpus of DA. Then they apply DL models built for MSA on that corpus of DA. They applied LSTM and BiLSTM on ASTD and LABR datasets, but the result wasn't promising as it was around 50%. After that, they applied them again to the Shami-Senti corpus. The results were better than before, as they exceeded 60%.

L.H. Baniata and S. Park proposed a DL model for Arabic SA. They combined CNN and BiLSTM once, and BiLSTM and CNN another time, and compared the results. They tested both models on the LABR dataset. The results have shown that CNN-BiLSTM outperformed the other one by an accuracy value of 86.43%.

Another research by Abdullah et al. [18] developed the SEDAT system, which is used for detecting sentiments and emotions written in the Arabic language. They applied CNN-LSTM with the help of document embedding on the SemEval 2018 dataset. They showed the performance results with Spearman correlation scores with a value of 81.80%.

Mohammed and Kora [11] proposed a corpus of Egyptian tweets consisting of 40k tweets with their polarity. Then they applied three DL models, which are CNN, LSTM, and CNN-LSTM.CNN achieved accuracy with a value of 75.72%, LSTM achieved 81.31%, and CNN-LSTM 88.05%. Also, they applied a data augmentation technique, which affects DL performance.

Despite the fact that most models of deep learning have enhanced the accuracy of Arabic sentiment analysis, there is still potential for development. The findings of this review indicate that more efforts are needed to develop DL models. This motivated us to investigate various deep learning models to improve the accuracy of Arabic SA.

III. Methods and Materials

This section shows the dataset that was used as well as the details of the proposed model.

A. Used Dataset

The dataset utilized is mainly drawn from the Corpus of Arabic Egyptian tweets [11], a corpus of 40,000 tweets written in the Egyptian dialect and modern standard Arabic (MSA). This corpus was built and labeled by Mohammed and Kora. It consists of positive and negative tweets of the same size as 20k tweets on several topics. Fig. 1 shows the tweet count along with their sentiment.

Mohammed and Kora constructed the corpus from clear, obvious tweets, which have a positive or negative sentiment. In addition, two independent experts were invited to check the annotation of the corpus to validate it.

B. Proposed Model

The proposed model is divided into four major phases: data pre-processing, tokenization, word embeddings, and different deep learning models, as shown in Fig. 2.

1) Preprocessing: Firstly, the preprocessing phase consists of two important main functions, which are data cleaning and preprocessing functions. The used corpus [10] is already filtered, cleaned, and tweets are labeled. Also, repeated hashtags, tweets, emojis, and non-Arabic letters are removed from tweets. So, preprocessing functions were applied directly on tweets like removing stop words, stemming and tokenizing.

Table III shows tweet examples before and after applying preprocessing functions. Stop words are words that usually appear in all tweets but are not important. These words should be deleted because they will not be distinguished when used as features in classification tasks. Stemming is the way suffixes are removed and words are reduced in their word stem.

2) Word embeddings: Word embeddings are a type of word representation that lets words with the same meanings be represented in the same way. Different approaches are available for word embeddings, such as Glove [21], created by Stanford, FastText [22], created by Facebook, and Word2Vec [23], created by Google. In general, there are two Word2vec models: a Continuous Word Bag (CBOW) and Skip-Gram (SG). The CBOW model, which predicts the current target word using context, although the SG predicts the context using a given word.

In such a case, the CBOW model is used for word vector representation. A pre-trained word2vec model is first implemented to generate the feature vectors of words, which will be used later as pre-trained vectors to generate the semantic vectors of words.

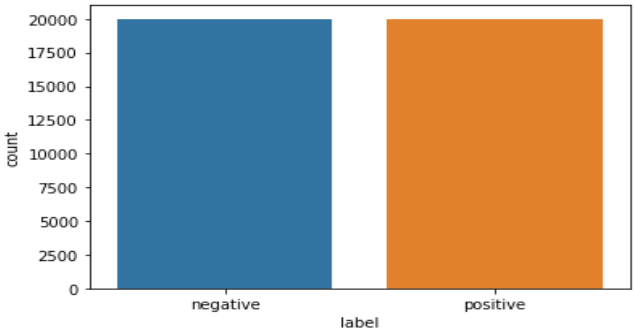

Fig. 1. Bar Chart of a Negative and Positive Count.

TABLE III. Examples for Pre-and Post-Preprocessing Tweets

pre-preprocessing Tweet	Post-processing tweet	Polarity
اكبر خطا ترتكبه ان تعامل الناس باخلاقك انت مش باخلاقهم هما . Translated as : The biggest mistake you make is to treat people with your morals, not theirs.	كبر خطا ركب عمل ناس خلق انت مش خلق	negative
دائما اكره اخر ليله في كل مكان . Translated as : I always hate the last night everywhere.	دما اكر اخر لبل كان	negative
احتاج صديق حقيقى يواسينى ويخفف عنى . Translated as : I need a true friend who comforts me and relieves me	حاج صدبق حقىقى بواسنى بخفف عنى	positive
لازم اتعلم الثبات الانفعالى زيهم كدا . Translated as :I have to learn emotional stability like that.	لزم علم ثبت على زبهم	positive

3) Model architecture: In this section, four deep learning models are proposed to detect the polarity of Arabic text as shown in Fig. 2. In particular, the proposed models are convolution neural network (CNN), long short-term memory (LSTM), a hybrid of (CNN-LSTM), and bidirectional LSTMs (Bi-LSTM).

a) CNN architecture model: The convolution neural network (CNN) is a popular unsupervised learning algorithm. The CNN model architecture is shown in Fig. 3. Using the word2vec model, the word embedding is generated. The first layer of CNN is the convolution layer, which has 32 filters along with Relu activation. A filer size equal to 8 is added with a filer size equal to 8 to extract the characteristics of the phrases. These filters convert the input and create characteristic maps (varying lengths). The second layer is the global maximum pooling layer that captures the most essential information from previous characteristics. The third layer, the Gaussian noise layer, is added to moderate overfitting, which works as a regularisation layer. A fully connected layer takes the generated features and pools them together to create the final predictions, which constitutes the fourth layer. A dropout layer is then added to the network to regularise it and prevent it from overfitting. The last layer of the sigmoid function type is a dense (completely linked) layer. This layer produces the network output, which classifies the input tweet as positive or negative.

b) LSTM architecture model: The long short-term memory (LSTM) unit is commonly constructed from a cell for memory and three gates to control the information flow to and from the cell over time. The three gates are called an input gate, an output gate and a forget gate. The LSTM model layers are shown in Fig. 4. The word embeddings are delivered to the cells of LSTM after applying the embedding layer. On these word embeddings, LSTM cells are trained and their prediction words are produced. A dropout layer is followed by a Gaussian noise layer to handle the overfitting by injecting noise during the time of training, and that reduces the computational effort during the time of testing. The words of the prediction are fully linked using a dense sigmoid layer.

c) CNN-LSTM model: This architecture was primarily named a Long-term Recurrent Convolutional Network or LRCN model, although the more generic name "CNN-LSTM" is used to refer to LSTMs that use CNN. The CNN extracts features from the sentences and represents them. however, LSTM works on extracted features where it takes the context and word ordering into consideration. Fig. 5 shows the CNN-LSTM architecture. It consists of an embedding layer that feeds into the convolution layer; after that, the output is considered as an input to the global max-pooling layer, followed by LSTM, followed by a dropout and Gaussian noise layer to regularise the output and prevent overfitting; and finally, a flattening layer with a sigmoid function that would give a negative or positive result.

d) BiLSTM architecture model: Generally, the BiLSTM is an expansion of regular LSTMs to increase model performance for sequence classification problems. This architecture is primarily known as the Bidirectional LSTM. Two LSTMs are trained in the input sequence instead of a single LSTM. The first is the input sequence, while the second is the input sequence is reversed. It can add to the network context and lead to faster and even more comprehensive learning of the problem. Fig. 6 shows the BiLSTM model layers. The embedded words are added after the use of the word embedding layer. Then apply BiLSTM to be trained on the embeddings of words and produce a set of word predictions that are linked with Gaussian dropout followed by a dense layer with a sigmoid function that would predict the sentiment of the result.

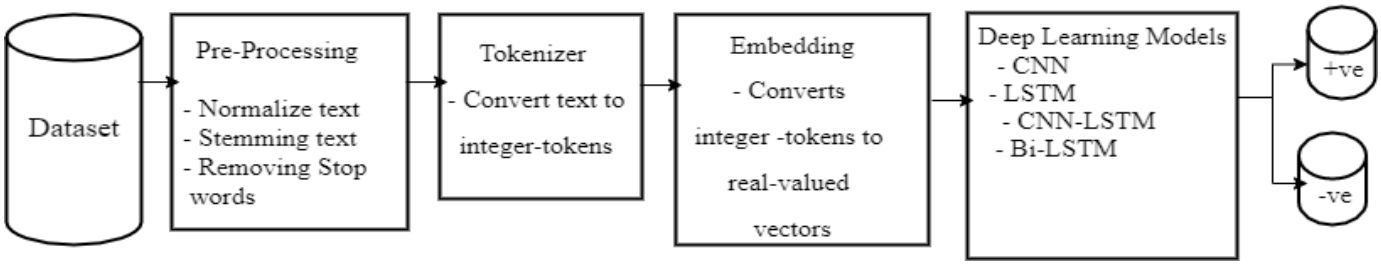

Fig. 2. Proposed Model Architecture.

Fig. 3. CNN Architecture Layers.

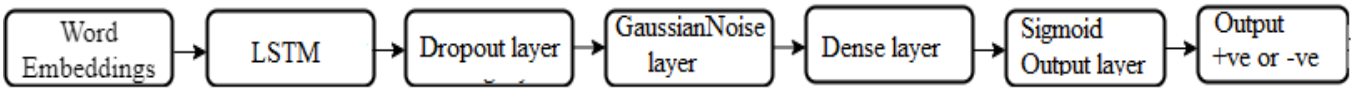

Fig. 4. LSTM Architecture Layers.

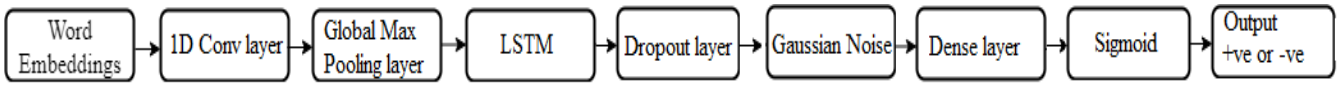

Fig. 5. CNN- LSTM Architecture Layers.

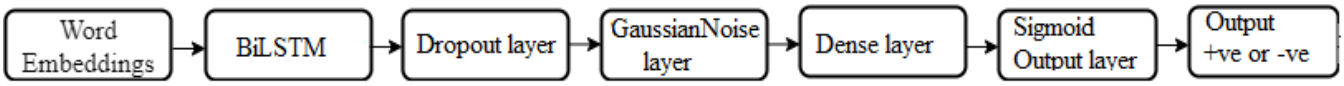

Fig. 6. BiLSTM Architecture Layers.

IV. Result and Discussion

The performance assessment of the proposed models is conducted using the most common evaluation metrics, which are accuracy, precision, recall, and F-measure. The (k-fold cross-validation) method is applied to the entire corpus [10] to evaluate the results. For the experimental setup, Google Colab Pro is used with a TPU hardware accelerator in Python 3.7.

A. Performance Evaluation Metrics

The confusion matrix is usually a type of matrix used to find the accuracy of classifiers. Each test data instance is assigned to an element, which belongs to a set of P, N comprising both positive, and negative class labels in the binary classification issue. There are four possible outputs given a model and an instance. (TP) the right number of tweets categorized as positive, (FP) the wrong number of tweets categorized as positive, (TN) is the correct number of tweets that are identified as negative, (FN) is the wrong number of tweets labeled as negative. A confusion matrix is created in the instances analysis of the given model and collection of test examples. Equation (1) displays the mathematical precision equation used to measure the true positive tweet, which indicates in Eq. (2) a part of positive tweets for a particular class. Equation (3) shows the mathematical equation for Accuracy. Equation (4) the F1score is determined by taking into account the recall and the precision of the test data.

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (1)$$

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

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FP+TN+FN)} \quad (3)$$

$$\text{F1score} = \frac{(2\times Percision\times Recall)}{(Percision+Recall)} \quad (4)$$

B. Experimental Result

The results of CNN, LSTM, CNN-LSTM, and BiLSTM are presented in this section. Each model was trained and tested using a data split (80%, 20%), bearing in mind that 10% is considered as a validation set for tuning the hyper-parameter. In addition, fivefold cross-validation is applied to improve performance, and each data division uses an average of five different runs. Given the equal distribution of the classes of positive and negative over the corpus, precision is a sufficient criterion for assessing the models. However, accuracy, recall, and f-score are also put into consideration as performance indicators for a proper interpretation of the findings.

Several experiments were conducted to enhance and achieve the best set of multiple hyper-parameters in each model. The optimal hyperparameter values utilized in the four models are shown in Table IV.

Table V displays the optimal CNN model combinations as well as the hyperparameters. The length of the sequence is 31, which is the biggest length of the tweet. In addition, 32 filters vary in 8 sizes of regions. Additionally, vocabulary size is determined by the number of words entered each time, and 10,000 words are selected.

In addition, the optimal settings in the LSTM model are shown in Table VI. In this setting, each layer of LSTM with 128 cells is used.

Table VII presents the best CNN-LSTM model configuration. This model is a hybrid of the previous two models in the same manner of settings except that the filter size is 16 instead of 8.

The BiLSTM settings are shown in Table VIII, which consists of one Bidirectional layer of LSTM using 128 cells.

TABLE IV. CNN, LSTM,CNN-LSTM and BiLSTM Hyperparameter Values

Learning rate	optimizer	Recurrent Dropout rate	Size of batch	# epochs
0.001	adam	0.5	32	100

TABLE V. CNN Setting Parameters

length of Sequence	#Filters	Filter size	Size of vocab
31	32	8	10,000

TABLE VI. LSTM Setting Parameters

LSTM size	LSTM layer	Recurrent dropout	Output dropout
128	1	20%	20%

TABLE VII. CNN-LSTM Setting Parameters

length of Sequence	31
# filters	32
Filter size	16
# LSTM layer	1
#LSTM cells	256
Recurrent dropout	20%
Output dropout	20%
Size of vocab	10,000

TABLE VIII. BILSTM SETTING PARAMETERS

LSTM cells	BiLSTM layer
128	1

Table IX shows the results of the proposed model experiments using the 5-fold cross-validation method. The results indicate that CNN achieved values of 99.65%, 99.78%, 99.77%, and 99.78% for accuracy, precision, recall, and F-measure respectively. LSTM achieved 91.83% for accuracy value, 90.97% for precision value, 90.97% for recall value and 90.98% for F-measure. Although CNN-LSTM achieved accuracy with a value of 73.19%, precision with a value of 74.02%, recall with a value of 74.02%, and the value of F-measure is 74.02%. While BiLSTM achieved values of 91.73%, 91.74%, 91.74%, and 91.74 in accuracy, precision, recall, and F-measure, respectively.

TABLE IX. RESULTS OF PROPOSED MODELS

Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
CNN	99.650	99.775	99.774	99.775
LSTM	91.830	90.974	90.973	90.975
CNN-LSTM	73.19	74.025	74.024	74.023
BiLSTM	91.730	91.740	91.735	91.741

C. Baseline and Evaluation

In this part, the model's performance is compared with the latest work existing in the literature which was introduced by Mohammed and Kora [11]. They introduced a labeled corpus consisting of 40k tweets in colloquial Arabic. They also applied three deep learning techniques to their corpus. So their work is considered our benchmark. For evaluation, we used their corpus [10] to achieve a fair comparison.

The research [11] applied CNN, LSTM, and CNN-LSTM as models. After that, they used the train/test split validation method to validate and test their models. They used three different test sizes, 30%, 40%, and 20%, respectively. Then they average the accuracy values for each data split.

In this paper, we propose another strategy. We apply the same models with core modifications to their structure and different hyper-parameters. also, an additional model is proposed named BiLSTM.

In the proposed CNN, another pooling layer is added called the Global Max Pooling layer. In this case, we set the pool size to the same as the input size. So it reduces the dimensionality of the feature maps output by the convolutional layer. Also, Gaussian noise is added via a separate layer named the GaussianNoise layer after the Global Max Pooling layer which makes CNN noise regularization. This layer has a regularising effect and reduces overfitting.

Also in the proposed LSTM, a GaussianNoise layer is added between an LSTM recurrent layer and a dense fully connected layer. In the same way, the layer is added in the proposed BiLSTM between the bidirectional LSTM layer and the dense layer.

To avoid sampling bias, we can think of a slightly different validation method. So, k-fold cross-validation was introduced. The data is divided into K folds. The data is trained and tested on the single fold which has been left out. This is done for all combinations and the results are averaged in each case. The advantage is that both training and validation are used by all observations, and every observation is validated once. Typically, we use k=5, because it is good enough to balance computational complexity and accuracy of validation.

The results in Table X show that the proposed CNN model outperformed by 23%, 25%, 12%, and 24% in terms of accuracy, precision, recall, and F-measure, respectively. Fig. 7 provides a comparison between CNN and the proposed CNN.

Also, Table XI shows that the proposed LSTM model outperforms by 10%, 10%, 9%, 9% in terms of accuracy, precision, recall, and F-measure respectively. Fig. 8 provides a comparison between LSTM and the proposed LSTM.

Contrary to expectation, the result of the proposed CNN-LSTM was not as high as the case with CNN and LSTM. The practical results showed the extent of convergence between the proposed model and the original model, as shown in Table XII.

To complete the experiments, BiLSTM was applied as an extra model which achieved accuracy with a value of 91.73% as shown in Table XII.

TABLE X. COMPARISON BETWEEN THE PROPOSED CNN MODEL AND MOHAMMED AND KORA. [11]

Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
CNN	75.72	74.60	78.03	75.06
Proposed CNN	**99.65**	**99.775**	**99.774**	**99.775**

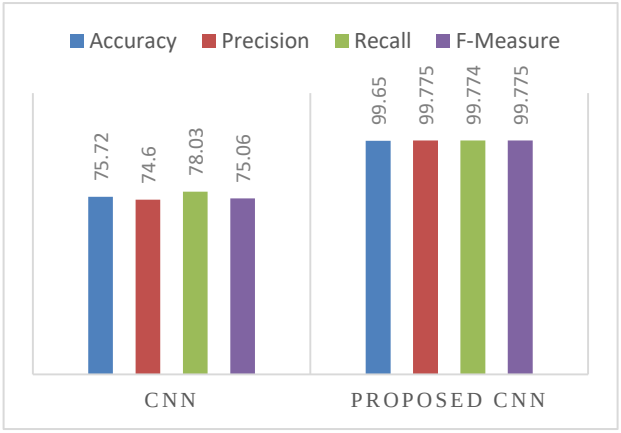

Fig. 7. CNN and Proposed CNN Comparison.

TABLE XI. COMPARISON BETWEEN THE PROPOSED LSTM MODEL AND MOHAMMED AND KORA [11]

Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
LSTM	81.31	80.92	81.99	81.25
Proposed LSTM	**91.830**	**90.974**	**90.973**	**90.975**

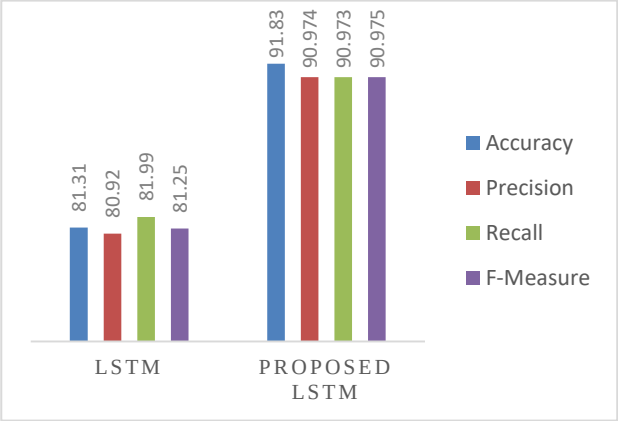

Fig. 8. LSTM and Proposed LSTM Comparison.

TABLE XII. COMPARISON BETWEEN THE PROPOSED MODEL AND MOHAMMED AND KORA [11]

Model	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
CNN -LSTM	78.46	80.27	77.38	77.86
Proposed CNN-LSTM	**73.19**	**74.025**	**74.024**	**74.023**
Proposed BiLSTM	**91.730**	**91.740**	**91.735**	**91.741**

V. CONCLUSION AND FUTURE WORK

This work focused on the problem of dialectal Arabic sentiment analysis using deep learning approaches. The proposed four deep learning techniques are tested on the used corpus. In particular, the proposed models are CNN, LSTM, and CNN-LSTM. Additionally, the experiments were extended by applying bidirectional LSTMs (BiLSTM). The obtained results show that the proposed CNN achieved an accuracy of 99.7%, the proposed LSTM achieved 91.83%, the proposed CNN-LSTM achieved 73.19%, and the proposed BiLSTM achieved 91.73%. The evaluation shows the higher performance of the proposed model in comparison with different models existing in the literature using the same corpus. Best results were achieved by using the combination of the Global Max Pooling layer and a GaussianNoise layer in the proposed CNN.

In future work, the CNN-LSTM model is planned to be improved so it will achieve better results. Also, we look forward to using other word representations such as Glove and FastText to see their effects on the results using different deep learning approaches. Additionally, we plan to use the data augmentation technique on the corpus to test its impact on corpus size and the performance of the deep learning approaches used.

REFERENCES

[1] K. Sarkar, "Sentiment polarity detection in Bengali tweets using deep convolutional neural networks," J. Intell. Syst., vol. 28, no. 3, pp. 377–386, 2019.

[2] R. Baly, R. Hobeica, H. Hajj, W. El-Hajj, K. B. Shaban, and A. Al-Sallab, "A meta-framework for modeling the human reading process in sentiment analysis," ACM Trans. Inf. Syst., vol. 35, no. 1, pp. 1–21, 2016.

[3] G. Badaro et al., "A survey of opinion mining in Arabic: A comprehensive system perspective covering challenges and advances in tools, resources, models, applications, and visualizations," ACM Trans. Asian Low-Resource Lang. Inf. Process., vol. 18, no. 3, 2019, doi: 10.1145/3295662.

[4] A. Hamdi, K. Shaban, and A. Zainal, "A review on challenging issues in Arabic sentiment analysis," J. Comput. Sci., vol. 12, no. 9, pp. 471–481, 2016, doi: 10.3844/jcssp.2016.471.481.

[5] A. Abdelwahab, F. Alqasemi, and H. Abdelkader, "Enhancing the Performance Of Sentiment Analysis Supervised Learning Using Sentiments Keywords Based Technique," in CS & IT Conference Proceedings, 2017, vol. 7, no. 1.

[6] E. Refaee and V. Rieser, "Benchmarking machine translated sentiment analysis for Arabic tweets," in Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop, 2015, pp. 71–78.

[7] A. Hamdi, K. Shaban, and A. Zainal, "A review on challenging issues in arabic sentiment analysis," 2016.

[8] I. Guellil, H. Saâdane, F. Azouaou, B. Gueni, and D. Nouvel, "Arabic natural language processing: An overview," J. King Saud Univ. - Comput. Inf. Sci., vol. 33, no. 5, pp. 497–507, 2021, doi: 10.1016/j.jksuci.2019.02.006.

[9] A. Dahou, S. Xiong, J. Zhou, M. H. Haddoud, and P. Duan, "Word embeddings and convolutional neural network for arabic sentiment classification," in Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers, 2016, pp. 2418–2427.

[10] R. Kora and A. Mohammed, "Corpus on Arabic Egyptian tweets." Harvard Dataverse, doi: doi:10.7910/DVN/LBXV9O.

[11] A. Mohammed and R. Kora, "Deep learning approaches for Arabic sentiment analysis," Soc. Netw. Anal. Min., vol. 9, no. 1, pp. 1–12, 2019.

[12] A. B. Nassif, A. Elnagar, I. Shahin, and S. Henno, "Jou," Appl. Soft Comput. J., p. 106836, 2020, doi: 10.1016/j.asoc.2020.106836.

[13] A. al Owisheq, S. al Humoud, N. al Twairesh, and T. al Buhairi, "Arabic sentiment analysis resources: A survey," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9742, no. 12, pp. 267–278, 2016, doi: 10.1007/978-3-319-39910-2_25.

[14] M. Attia, Y. Samih, A. Elkahky, and L. Kallmeyer, "Multilingual Multi-class Sentiment Classification Using Convolutional Neural Networks," pp. 635–640, 2016.

[15] M. Heikal, M. Torki, and N. El-Makky, "Sentiment analysis of Arabic Tweets using deep learning," Procedia Comput. Sci., vol. 142, pp. 114–122, 2018.

[16] A. Elnagar, L. Lulu, and O. Einea, "ScienceDirect ScienceDirect An Annotated Huge Dataset for Standard and Colloquial Arabic Reviews for Subjective Sentiment Analysis," Procedia Comput. Sci., vol. 142, pp. 182–189, 2018, doi: 10.1016/j.procs.2018.10.474.

[17] H. Abdellaoui and M. Zrigui, "Using Tweets and Emojis to Build TEAD : an Arabic Dataset for Sentiment Analysis," vol. 22, no. 3, pp. 777–786, 2018, doi: 10.13053/CyS-22-3-3031.

[18] M. Abdullah, M. Hadzikadic, S. Shaikh, and N. Carolina, "SEDAT : Sentiment and Emotion Detection in Arabic Text using CNN-LSTM Deep Learning," 2018, doi: 10.1109/ICMLA.2018.00134.

[19] K. Abu and C. Simon, "Can Modern Standard Arabic Approaches be used for Arabic Dialects ? Sentiment Analysis as a Case Study," 2018.

[20] L. H. Baniata and S. Park, "Sentence Representation Network for Arabic Sentiment Analysis," pp. 470–472, 2016.

[21] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[22] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of Tricks for Efficient Text Classification," arXiv Prepr. arXiv1607.01759, 2016.

[23] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," arXiv Prepr. arXiv1310.4546, 2013.