A Deep Learning Approach Combining CNN and Bi-LSTM with SVM Classifier for Arabic Sentiment Analysis

Omar Alharbi
Computer Department
Jazan University, Jazan, Saudi Arabia

***Abstract*—Deep learning models have recently been proven to be successful in various natural language processing tasks, including sentiment analysis. Conventionally, a deep learning model's architecture includes a feature extraction layer followed by a fully connected layer used to train the model parameters and classification task. In this paper, we employ a deep learning model with modified architecture that combines Convolutional Neural Network (CNN) and Bidirectional Long Short-Term Memory (Bi-LSTM) for feature extraction, with Support Vector Machine (SVM) for Arabic sentiment classification. In particular, we use a linear SVM classifier that utilizes the embedded vectors obtained from CNN and Bi-LSTM for polarity classification of Arabic reviews. The proposed method was tested on three publicly available datasets. The results show that the method achieved superior performance than the two baseline algorithms of CNN and SVM in all datasets.**

***Keywords*—Sentiment analysis; Arabic sentiment analysis; deep learning approach; convolutional neural network CNN; bidirectional long short-term memory Bi-LSTM; support vector machine; SVM**

I. Introduction

Sentiment analysis (SA) is one of the most active research areas in natural language processing (NLP). The SA task aims to equip a machine with the ability to categorize people opinions into positive or negative based on the sentiment expressed in the texts. SA is technically a challenging task, as human language is complex and diverse. Nevertheless, research on SA has achieved considerable progress, especially for the English language, while Arabic SA is developing slowly despite increasing Arabic language usage on the Internet [1]. That can be attributed to the Arabic language's complex morphological nature and diverse dialects [2]. Additionally, technical issues such as scarce linguistic resources and limited linguistic tools of Arabic increased the level of complexity [3]. The Arabic language is one of the most common languages that is used by more than 400 million[1] individuals to daily communication. Arabic is formally written in a form called Modern Standard Arabic (MSA), which is understood all over the Arabic world. Despite this fact, internet users usually tend to use dialectal words alongside MSA to write reviews, tweets, or comments. Dialectal words result from behaviors such as replacing characters and changing the pronunciation or the style of writing of nouns, verbs, and pronouns of MSA [4]. Thus, there are no standard rules that can handle the morphological or syntactic aspects of reviews written in MSA with the presence of dialects.

One of the prominent approaches employed in SA is machine learning (ML). In this context, SA is formalized as a classification problem which is addressed by using algorithms like Naive Bayes (NB), Support Vector Machine (SVM) and Maximum Entropy (MaxEnt). In fact, ML has proved its efficiency and competence; therefore, in the past two decades, it mostly dominates the SA task. However, a fundamental problem that would decay the performance of ML is the text representation through which the features are created to be fed to the learning process. For text representation, the most used model in literature is the bag-of-words (BOW) with unigram, bigram, or part of speech (POS). By using BOW, words are often weighted by their presence (binary scheme) or frequency like in term frequency (TF) or term frequency-inverse document frequency (TF-IDF). This model is relatively straightforward and important; however, it can be problematic and may degrade ML-based sentiment classification performance when it comes to a large number of features or the need for semantic information [5-8].

Over the past decade, the deep learning approach and word vector representations have attained an increasing interest from NLP researchers as they can successfully handle the limitations of traditional ML methods at NLP tasks, including SA (Kim 2014). Deep learning is an extended approach of ML and a subset of the neural network. It has a structure composed of multiple hidden layers, which enable it to automatically discover semantic representations of texts from data without feature engineering [9]. This approach has improved the state-of-the-art in many SA tasks, including sentiment classification, opinion extraction, and fine-grained sentiment analysis [10]. Along with deep learning, word vector representations, also known as word embeddings, has emerged as a powerful features representation model for the classification task. Word embeddings is a modern approach to learn real-valued low-dimensional vector space representations for a text [11]. It aims to encode continuous semantic similarities between words based on their distributional properties in a large corpus [12]. Word embeddings are typically extracted by using neural networks as the underlying predictive model [13]. One of the most used methods for word embedding is Word2Vec that was proposed by [11]. Word2Vec produces useful word representations learned by a 2-layer neural network based on

[1] https://en.unesco.org/commemorations/worldarabiclanguageday

the model that has been proposed in [14]. Word2vec is one of the key methods, which has led to the state-of-the-art results achieved by the deep learning approach on SA. Word2vec can be implemented using two main learning models; they are continuous bag-of-words (CBOW) and Skip-gram.

Researchers have proposed many variants of deep learning models to address the SA of English, which have shown promising efficacy. For instance, in [15], the authors use Recursive Neural Tensor Network (RNTN) along with word embeddings to address fine-grained sentiment classification for phrase level. In another work [16], the authors apply an ensemble model that includes Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) networks on top of pre-trained word vectors for tweets sentiment classification. The authors in [17] develop an adaptive Recursive Neural Network (RNN) that uses more than one composition functions and adaptively selects them depending on the linguistic tags and the combined vectors for tweets sentiment classification. Unlike English, there have not been many studies that address the Arabic language in SA based on a deep learning approach and word embeddings, except a few such as [18-22], which will be discussed with other studies in the following section. Accordingly, there is still room for exploring different deep learning algorithms and investigating their performance in addressing Arabic SA, especially with the challenges that the Arabic language is still imposing.

In this work, we propose a new method that combines deep learning models with SVM for improving the SA of Arabic reviews. Conventionally, a deep learning model's architecture includes features extraction layers (e.g. CNN or RNN) followed by a fully connected layer used for training the model parameters and classification task. However, in this work, we decided to modify the structure and explore the impact of this structure on sentiment classification performance. The structure is altered by replacing the fully connected network with a linear SVM classifier that trains the embedded vectors generated by the combined deep learning models. This is justified by SVM's remarkable improvement in recent research in SA, including Arabic SA, such as in [20, 23, 24]. The idea of combining the neural network and SVM was initially proposed in [25] to improve multilayer perceptron classifiers. In this respect, many researchers presented various models for different classification tasks including image classification [26, 27], visual recognition [28], intrusion detection [29], object categorization [30], speech recognition [31], and sentiment analysis [32]. Although the proposed method is inspired by these studies, it is different as we use a linear SVM as a replacement for the fully connected layer instead of replacing only the top layer's activation function (i.e. Softmax or Logistic). In particular, this research presents a combination of CNN and Bidirectional-LSTM with linear SVM to improve the sentiment classification performance on Arabic reviews at the document level. To the best of our knowledge, this is the first attempt to use such a combination for Arabic SA. In this work, we used three publically available datasets for evaluation: LABR [33], OMCCA [34], and the dataset presented in [35].

The remaining sections of this paper are organized as follows: Section 2 discusses the related work in Arabic SA. In Section 3, we present the proposed method. Section 4 describes the experimental details used to evaluate the performance of the proposed method. In Section 5, we report and discuss the obtained results. Section 6 presents the conclusion and future work.

II. Related Work

Researchers have presented several methods and models to tackle Arabic SA based on three major approaches, namely, machine learning (ML) [36, 37], semantic orientation [38, 39], and hybrid approach [40]. In this research, we focus on the studies that employ the ML approach, specifically deep learning methods, which have been increasingly utilized in the past decade. These methods have presented remarkable improvements in the SA field, especially for the English language, such as in [41-45]. On the other hand, only a few studies have attempted to utilize the deep learning approach for Arabic SA. For instance, in [46], the authors use an ensemble model of CNN and LSTM proposed by [16] to predict the sentiment of Arabic tweets for the sentence level. The model is trained on top pre-trained word vectors developed in [47]. They evaluated the model on ASTD dataset [48] and achieved an F1-score of around 64% and an accuracy of around 65%. Another work applies CNN and LSTM to sentiment analysis of Arabic tweets described in [49]. They designed a system to identify the sentiment's class and intensity as a score between 0 and 1. The authors used word and document embedding vectors to represent the tweets. They translated the tweets into English to benefit from the available preprocessing tools. As they highlighted, the step of the translation led to degrading the overall performance. Although the system includes some preprocessing steps, it excludes some other important normalizing processes such as removing diacritics, punctuations and repeating characters.

In [50], the authors investigate LSTM and Bi-LSTM models' performance on Saudi dialectical tweets. After some basic normalization processes, they use the CBOW model to represent words with vectors. As reported, Bi-LSTM outperformed LSTM and SVM with an accuracy of 94%. The study of [19] explores different deep learning models, including a combination of LSTM and CNN, to predict tweets' sentiment. They also investigate the use of CBOW and Skip-gram models of available pre-trained word vectors introduced in [51]. The experiments show that the highest results were obtained by using a two-layer LSTM model. Barhoumi et al. [52] utilized Bi-LSTM and CNN to evaluate different types of Arabic-specific embeddings. They suggested using available Arabic NLP tools to address the effect of the agglutinate and morphological rich specificity of Arabic on word embedding quality. However, these NLP tools such as lemmatization, light stemming, and stemming have been mostly developed to MSA texts.

The work in [53] reports efforts to detect the sentiment of tweets of time series data in the domains of stock exchange and sport. They use a deep neural network model with eight fully connected hidden layers in which each layer has 100 neurons. They compared the model against KNN, NB, and decision tree classifiers. Their model showed superiority with an F1-sore of around 91% and accuracy of around 88%. Nevertheless, the authors did not mention the kind of feature representation used

in this work. The work in [54] explores different architectures for Arabic sentiment classification using Deep Belief Networks (DBF), Deep Auto Encoders (DAE), and Recursive Auto Encoder (RAE). The experimental results show that RAE obtained the best performance due to its capability to handle the sentence's context and order of parsing. Yet, the authors focus on sentence-level sentiment classification, and they used the traditional BOW to represent texts. In [55], the authors implemented LSTM to handle Arabic sentiment classification. The model was implemented along with character/word embedding features. In their experimental results, LSTM has shown the ability to improve the sentiment classification and achieved an accuracy of around 82%. The same author in another work [56] found that SVM work better than RNN on the same dataset, but the execution time of RNN was faster. However, both studies focus on aspect-based sentiment analysis of Arabic reviews.

This current work introduces a combination of CNN with Bi-LSTM and a linear SVM classifier for Arabic sentiment classification. Unlike the studies mentioned above, the proposed model does not follow the conventional CNN and Bi-LSTM. Rather, the fully connected layers would be replaced with a linear SVM algorithm to perform the classification task.

III. Proposed Method

The proposed method for sentiment classification of Arabic reviews starts with preparing data by applying some data preprocessing techniques. Then, the word vector representation is built using pre-trained word embeddings trained on other large Arabic text corpora. After that, we combine CNN and Bi-LSTM for generating the final features representation to be passed to a linear SVM classifier.

A. Data Preprocessing

The preprocessing stage consists of removing noise from data and normalization. The process of removing noise from data includes removing repeated letters, diacritics, punctuations, numerals, English words, and elongation. After that, a normalization process was applied to particular letters, for example, the letters (آ, إ, أ) were converted to (ا), the letters (ى, ئ) were converted to (ي), the letter (ة) was converted to (ه), and finally, the letter (ؤ) was converted to (و). It is worth mentioning that we implemented these particular preprocessing steps to be compatible with those used in the pre-trained word embeddings model we intend to use. More details about the pre-trained word embeddings model in the next section.

B. Word Vector Representations

To generate representations for the reviews, we adopted the word vectors approach. Word vectors (also known as word embeddings) are vectors of real numbers representing the distribution of words or phrases in a given text [14]. This approach's key benefit is that high-quality word embeddings can be learned efficiently, mainly if a much larger corpus of text were used for training [57]. However, the adopted dataset's size is relatively small in our case, which might lead to low-quality word embeddings. Therefore, we decided to use a pre-trained word embeddings model that trained on other large Arabic texts. One of the most common Arabic pre-trained word embeddings is Aravec [47]. This pre-trained model was trained on texts from multiple sources such as Twitter, Web pages, and Wikipedia with tokens of more than 3,300,000,000. This model was implemented by the well-known word2vec technique described in [11] based on CBOW and Skip-gram architectures.

C. Convolutional Neural Network Layer

We first build a CNN layer to extract the features of reviews. The inputs to the CNN are matrices of word embeddings X that reflect the word vector representations of the reviews. Each review is mapped to a matrix of size $s \times d$, where s is the number of words in the review and d is the dimension of the embedding space. All reviews must be zero-padded to have the same matrix dimension $X \in \mathbb{R}^{s' \times d}$. Then, in order to compute a new feature map, inputs are convolved with a filter matrix $w \in \mathbb{R}^{h \times d}$, where h is the size of the convolution. The convolution is computed as in Equation 1:

$$c_i = f\left(\sum_{j,k} w_{j,k} (X_{[i:i+h-1]})_{j,k} + b\right) \quad (1)$$

Where $b \in \mathbb{R}$ is a bias term and $f(x)$ is a nonlinear activiation function. The output is the result of element-wise product between an input matrix X and a filter matrix w, which then be summed as a single value in a feature map $c = [c_1, c_2, \dots, c_n]$, where $c \in \mathbb{R}^{n-h+1}$. To obtain rich features representation the model can use multiple filters of different length that can work in parallel. Then, the convolved results are passed to the pooling layer to reduce the representation dimensionality by identifying the most important features. We employed the max-pooling technique which returns the maximum k values from the feature map $\hat{c} = \max(\boldsymbol{c})$. This technique has the ability to force the network to capture the most useful local features produced by the convolutional layer [58]. A typical architecture of a CNN layer is illustrated in Fig. 1. The output of this layer is fed into the next layer that is Bi-LSTM, based on the assumption that is more informative representation will be obtained, more details in the next section.

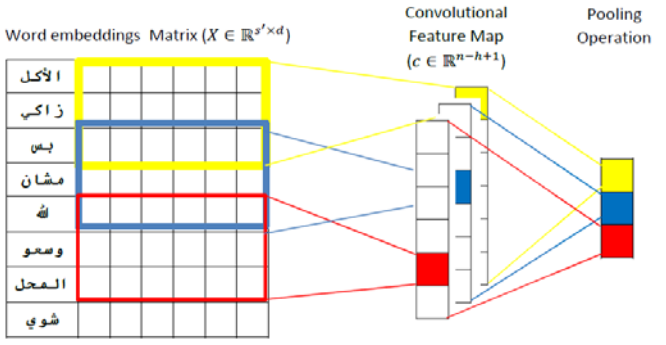

Fig. 1. A typical CNN Layer Architecture, the Colored Boxes Represent the Size of a Filter Spatially.

D. Bidirectional-Long Short-Term Memory (Bi-LSTM) Layer

As mentioned above the input into this layer is the final feature map obtained from CNN. First, let us present a little background about LSTM to understand the Bi-LSTM. LSTM is a type of recurrent neural network RNN, which was developed by [59]. The main objective of LSTM is to avoid vanishing gradient problem that encounters RNN, where the

gradient that is propagated back through the network either decays or grows exponentially over time [60]. In other words, RNN is not an effective model for understanding the context behind an input with long-term dependencies. On the other hand, LSTM with its complicated dynamics that consists of several so-called memory blocks, can effectively resolve this issue and make a prediction based on time series data. Each memory block is composed of an input gate i, a forget gate f, an output gate o, and a cell state c. These elements are used to compute the hidden state h by the following composite function in Equation 2:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)$$
$$c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)$$
$$h_t = o_t \tanh(c_t) \quad (2)$$

Where x_t is the input vector to the memory cell at time t, σ is the sigmoid function, W is the weight matrices, and b is the bias.

However, one downside of LSTM inherited from RNN is that it does not consider the whole context because the sentence is only read in a forward direction [16]. In this case, the context information provided by the future words will be dismissed, resulting in low classification performance. An extension of LSTM called bidirectional LSTM (Bi-LSTM) [61] was proposed to avoid this problem. In this work, we adopted this method to obtain more informative features based on the features given by CNN. Bi-LSTM simultaneously trains two separate LSTMs of opposite directions (one for forward and one for backward) on the input sequence and then merge the outputs. More formally, as explained above, the input vectors x_t are fed one at a time t into the LSTM, making use of all the available input information up to the current time frame t_c to predict y_{t_c}. However, applying the bidirectional network's notion will enable the network to use the input information coming up later than t_c by delaying the output by a certain number M time frames up to x_{t+M} to predict y_{t_c}. The structure of Bi-LSTM is illustrated in Fig. 2 that shows the basic structure of folded Bi-LSTM which computes the forward hidden sequence $\vec{h}$, the backward hidden sequence $\overleftarrow{h}$, and the output sequence y by iterating the backward and forward layers. This layer also if followed by a max-pooling layer as outlined in the previous section.

After this layer, we have a layer that concatenates the list of outputs from the previous layers into a single vector. Conventionally, this vector is passed to a fully connected layer that applies weights over the generated features $\alpha(w * x + b)$ to predict the class of a given input, where $w \in \mathbb{R}^{m \times m}$ is the weights matrix, and α is the activation function. However, in this work, we will replace these fully connected layers with a linear SVM to perform the prediction process, more details in the next section.

E. Support Vector Machine (SVM) Classifier

As mentioned earlier, we present a model that combines CNN and Bi-LSTM with SVM, where CNN and Bi-LSTM used for feature vectors generation and SVM for sentiment classification. SVM was employed as it has proven to be an effective algorithm in many NLP tasks, including SA. The key idea behind combining these heterogeneous methods is to use each method's advantages to improve the Arabic language's sentiment classification. In general, SVM is a machine learning algorithm for binary classification problems introduced by [63]. The SVM classifier aims to find an optimal hyperplane that classifies given data points into one of two classes by maximizing the margin. More formally, a linear SVM will receive a concatenated vector of features $x_i$associated with its labels $y_i$to be processed by a hyperplane $f(w, x) = w^T . x_i + b$ to compute the coefficients of the hyperplane as feature weights, where w is the normal vector to the hyperplane and also known as the weight vector and b is the bias.

The architecture of the proposed model is illustrated in Fig. 3. Obviously, our model's architecture is essentially the same as the basic CNN and Bi-LSTM with a difference in using the SVM algorithm for classification. Further clarity, CNN or Bi-LSTM typically use a fully connected network with different activation functions (i.e. logistic) for learning and classification, as explained earlier. However, in this work, we replace the fully connected layer with a linear SVM which use a margin-based loss function for classification.

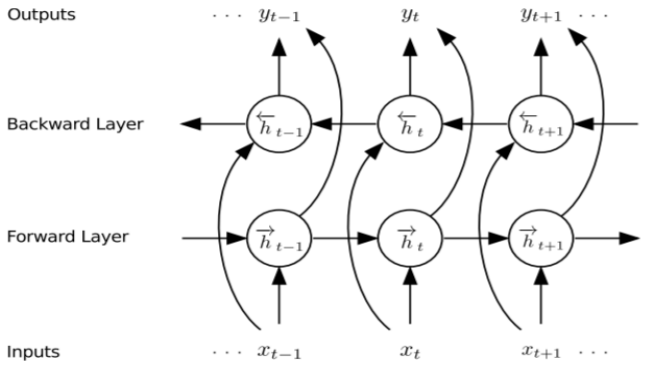

Fig. 2. Architecture of Bi-LSTM (Source: [62]).

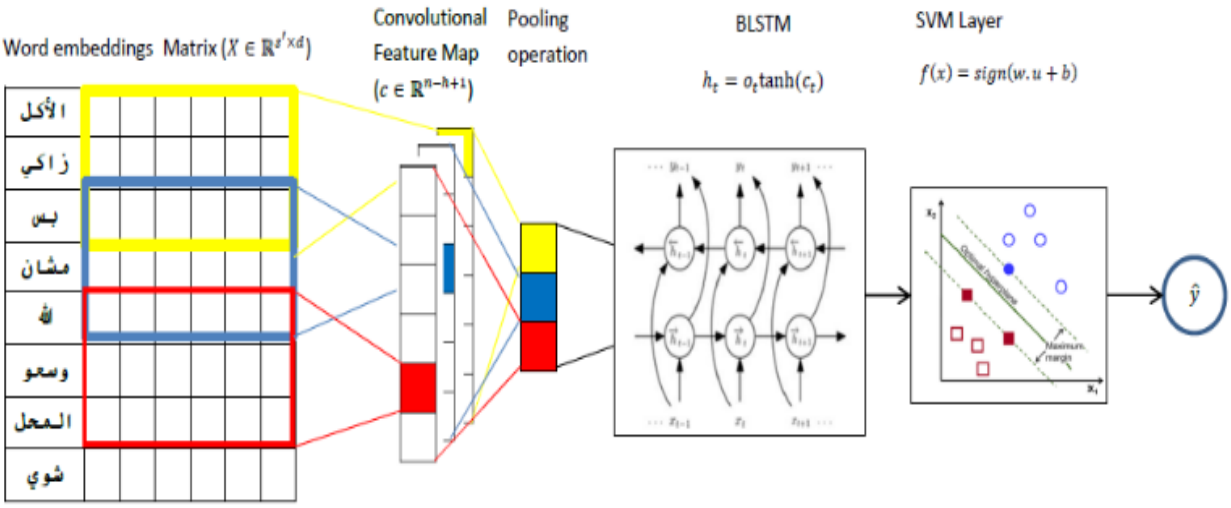

Fig. 3. Architecture of the Proposed Model.

IV. Experiments

This section presents the datasets, models settings, and evaluation metrics. For evaluation, we developed some experiments to compare our model's performance versus baseline models. Particularly, the author used the individual classifiers of CNN and SVM as baseline models. Additionally, we compared the proposed model results with some of the state-of-the-art deep learning models proposed by other related studies. The experiments also explore the best pre-trained word embeddings architecture for the proposed model. For this purpose, each performance experiment is carried out with CBOW and Skip-gram. To perform all the experiments in this work, we used the Keras library. More details about the experiments are provided in the following subsections.

A. Datasets

We used three publicly available datasets, namely, LABR, OMCCA, and the dataset presented in [35]. These datasets vary in size and writing form, where MSA and different Arabic dialects were detected. For example, LABR contains over 63000 reviews written mostly in MSA with the presence of dialectal phrases. On the other hand, On the other hand, OMCAA consists of over 28000 reviews written mostly in Jordanian and Saudi Arabic dialects. The last dataset presented 2400 reviews written in Jordanian dialect with the presence of MSA. All the datasets are annotated on the document level, and we considered only two polarity classes, which are positive and negative. The training/validation/testing split is set to 70%, 15%, 15%, respectively, in our experiments. Due to the limited capabilities of the device used to perform the experiments, we considered only the randomly balanced versions of these datasets. Table I presents more details about datasets.

TABLE I. Characteristics of Datasets used in our Experiments

Dataset	Balanced	Review Type
LABR	16444	Books
OMCCA	4990	Products
[35]	2400	products

B. Evaluation Metrics

The performance is quantified using the following evaluation metrics: F1-score, Precision, and Recall; see Equations 3, 4 and 5.

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

$$F1 - score = 2\frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

where TP, TN, FP FN indicate a true positive, true negative, false positive, false negative, respectively.

C. Hyper-parameters Settings

This section describes the hyper-parameters settings for each model used in the experiments. The process of selecting the optimal hyper-parameters is a challenging task, and it varies based on the characteristics of the dataset. To this end, we performed several trials to choose the hyper-parameters with which the classifiers may yield the best performance results. As a result, the hyper-parameters in Table II have been assigned to the layers of CNN and Bi-LSTM. It is worth mentioning that all the data points have been zero-padded before being passed to the deep learning models, so all the input vectors have the same length. In the case of the CNN is a baseline model, the word vectors will be fed to a fully connected layer of size 64 nodes. The dense hidden layer is trained and regularized using a Relu function and l_2-norm method, respectively. The weights are then passed to an output layer with a sigmoid function to give the final classification probability. We also applied a Dropout layer after each model and early stopping strategy to reduce the over-fitting problem. Then, we added a max-pooling layer after each layer to keep the most important features. During the networks' training process, Adam algorithm) is employed to perform the optimization with a learning rate of 0.001, and binary cross-entropy function for loss minimizing. A linear SVM classifier with its default parameters in Keras library has been employed for the classification process. Regarding generating word

embeddings based on Aravec, after many trials, we decided to apply the skip-gram and CBOW models built on Web pages with tokens of 132,750,000 and embedding dimension of 300. For words that are not contained in the pre-trained model, the embedding is set to a vector of zeros.

TABLE II. HYPER-PARAMETERS OF CNN AND BI-LSTM

Parameter	CNN	Bi-LSTM
Filters	250	-
Kernel size	4	-
Strides	3	-
Output dimensions	-	150
Activation	Relu	Tanh
Dropout	0.5	0.5
Max Pooling	2	2
Epoch size	30	30
Batch size	30	30
Loss function	binary_crossentropy	binary_crossentropy
Optimizer	Adam	Adam
L2 regularization	0.01	0.05

V. RESULTS

Results of the proposed model against the baseline models on the datasets are summarized in Table III. As it can be seen, our model with skip-gram and CBOW outperforms the other models in all datasets. The results show a significant improvement in the F1-score of the proposed model in all datasets, where the highest is achieved in the third dataset with around 8% compared to CNN and SVM when skip-gram is used and around 7% when CBOW is applied. Although the proposed model outperforms SVM and CNN with CBOW on OMCCA, the highest results are obtained using skip-gram with an improvement of 7% and 8% compared to CNN and SVM, respectively. It is also noticed that the proposed model show improvement on the LABR dataset when skip-gram is used with around 6% and 7% compared to CNN and SVM, respectively, and around 5% for both models in the case of skip-gram.

Additionally, the proposed model performs very well in terms of the recall and precision in all dataset with a bit of trade-off, where the recall is the highest compared to all models used with 91.6% on the third dataset. On the other hand, the highest precision value is around 91% and achieved by CNN on the OMCCA dataset. Based on the two-word embeddings models, CNN achieved classification performance close to SVM with slight superiority to the former. This might indicate that the representation captured by both models is not enough to identify the latent connections between words, and a richer representation is required for better performance.

Finally, after the proposed model has been proven to be effective in Arabic sentiment classification for all datasets used in this work, we compare our results with other studies' results that employed state-of-the-art deep learning models. We choose work that used LABR dataset as it is the most common dataset in the Arabic SA literature. To guarantee a fair comparison, we only consider those who experimented on balanced class labels and used Aravec pre-trained model. In this sense, the work in [64] reported a significant improvement in Arabic sentiment classification based on a combination of Bi-LSTM and CNN on different datasets, including LABR. However, our proposed model outperforms their model, where they achieved an F1-score of 76.9%, which is 10% less than the highest results of our model on LABR.

TABLE III. RESULTS OF A COMPARISON BETWEEN THE PROPOSED AND BASELINE MODELS ON THE USED DATASETS WITH SKIP-GRAM AND CBOW EMBEDDINGS MODELS

Dataset	Model	Representation	Recall	Precision	F1-score
LABR	CNN	Skip-gram	73.05	89.24	80.34
		CBOW	76.68	82.68	79.56
	SVM	TF-IDF	74.46	86.49	79.08
	Proposed Model	Skip-gram	85.92	86.36	**86.15**
		CBOW	82.0	86.31	84.10
OMCAA	CNN	Skip-gram	74.70	91.30	82.17
		CBOW	76.28	87.72	81.60
	SVM	TF-IDF	83.26	79.06	81.11
	Proposed Model	Skip-gram	90.0	89.28	**89.64**
		CBOW	88.35	84.29	86.27
[35]	CNN	Skip-gram	74.16	90.81	81.65
		CBOW	77.27	85.53	81.19
	SVM	TF-IDF	75.83	88.77	81.58
	Proposed Model	Skip-gram	91.66	86.61	**89.06**
		CBOW	88.33	86.88	87.60

VI. Conclusions and Future Work

This paper presented a model that uses a linear SVM classifier on top of a combination of CNN and Bi-LSTM for Arabic sentiment classification. Unlike the conventional architecture of the deep learning model, the proposed model was tailored by replacing the fully connected layer with an SVM classifier, which receives the embedded vectors extracted by CNN and Bi-LSTM. The experimentation has shown the effectiveness of the proposed model with a significant improvement over the baseline models. Furthermore, we showed that the proposed model outperforms one of the state-of-the-art deep learning models with a considerable improvement. Yet, there is room for improvement as the proposed model does not concern with some issues that might affect the performance (e.g. negation handling). Further work is also required to explore the effectiveness of the proposed model with deeper layers and diverse architectures on different Arabic benchmark datasets. Additionally, to extend the work so that it can be applied to other Arabic SA tasks such as aspect-based sentiment analysis.

References

[1] M. Al-Ayyoub, et al. A Comprehensive Survey of Arabic Sentiment Analysis. Information processing & management. 56(2), pp. 320-342, 2019.

[2] I. A. Al ‐ Sughaiyer and Al ‐ Kharashi, I. A. Arabic Morphological Analysis Techniques: A Comprehensive Survey. Journal of the American Society for Information Science and Technology. 55(3), pp. 189-213, 2004.

[3] S. R. El-Beltagy and Ali, A. "Open Issues in the Sentiment Analysis of Arabic Social Media: A Case Study," in 9th International Conference on Innovations in Information Technology. IEEE. Abu Dhabi, pp. 215-220, 2013.

[4] K. Shaalan, Bakr, H., and Ziedan, I. "Transferring Egyptian Colloquial Dialect into Modern Standard Arabic," in International Conference on Recent Advances in Natural Language Processing (RANLP–2007), Borovets, Bulgaria. pp. 525-529, 2007.

[5] A. L. Maas, et al. "Learning Word Vectors for Sentiment Analysis," in Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies. Association for Computational Linguistics. Portland, Oregon, USA, pp. 142-150, 2011.

[6] Y. Kim. "Convolutional Neural Networks for Sentence Classification," in Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics. Doha, Qatar, pp. 1746–1751, 2014.

[7] D. Tang, et al. "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification," in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics. Baltimore, Maryland, pp. 1555-1565, 2014.

[8] F. Xu, Zhang, X., and Yang, A. Investigation on the Chinese Text Sentiment Analysis Based on Convolutional Neural Networks in Deep Learning. CMC-Computers, Materials & Continua. 58(3), pp. 697–709, 2019.

[9] D. Tang, Qin, B., and Liu, T. Deep Learning for Sentiment Analysis: Successful Approaches and Future Challenges. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 5(6), pp. 292-303, 2015.

[10] L. Deng and Liu, Y. Deep Learning in Natural Language Processing (1st. ed). Springer. Singapore, 2018, DOI: 10.1007/978-981-10-5209-5.

[11] T. Mikolov, et al. "Efficient Estimation of Word Representations in Vector Space," in 1st International Conference on Learning Representations. Arizona, USA, pp. 2013.

[12] J. Turian, et al. "A Preliminary Evaluation of Word Representations for Named-Entity Recognition," in NIPS Workshop on Grammar Induction, Representation of Language and Language Learning. pp. 1-8, 2009.

[13] O. Levy and Goldberg, Y. "Neural Word Embedding as Implicit Matrix Factorization," in Advances in neural information processing systems. pp. 2177-2185, 2014.

[14] Y. Bengio, et al. A Neural Probabilistic Language Model. Journal of machine learning research. 3(Feb), pp. 1137-1155, 2003.

[15] R. Socher, et al. "Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank," in Proceedings of the 2013 conference on empirical methods in natural language processing. pp. 1631-1642, 2013.

[16] M. Cliche. "Bb_Twtr at Semeval-2017 Task 4: Twitter Sentiment Analysis with Cnns and Lstms," in Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). Association for Computational Linguistics. Vancouver, Canada, pp. 573–580, 2017.

[17] L. Dong, et al. "Adaptive Recursive Neural Network for Target-Dependent Twitter Sentiment Classification," in Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers). pp. 49-54, 2014.

[18] A. Dahou, et al. "Word Embeddings and Convolutional Neural Network for Arabic Sentiment Classification," in Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers. pp. 2418-2427, 2016.

[19] S. Al-Azani and El-Alfy, E.-S. M. "Hybrid Deep Learning for Sentiment Polarity Determination of Arabic Microblogs," in International Conference on Neural Information Processing. Springer. pp. 491-500, 2017.

[20] A. M. Alayba, et al. "Arabic Language Sentiment Analysis on Health Services," in 2017 1st International Workshop on Arabic Script Analysis and Recognition (ASAR). IEEE. pp. 114-118, 2017.

[21] R. Baly, et al. A Sentiment Treebank and Morphologically Enriched Recursive Deep Models for Effective Sentiment Analysis in Arabic. ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP). 16(4), pp. 23, 2017.

[22] A. M. Alayba, et al. "Improving Sentiment Analysis in Arabic Using Word Representation," in 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR). IEEE. pp. 13-18, 2018.

[23] A. Shoukry and Rafea, A. "Sentence-Level Arabic Sentiment Analysis," in the International Conference on Collaboration Technologies and Systems. IEEE. Colorado, pp. 546-550, 2012.

[24] O. Al-Harbi. Classifying Sentiment of Dialectal Arabic Reviews: A Semi-Supervised Approach. International Arab Journal of Information Technology. 16(6), pp. 995-1002, 2019.

[25] J. A. Suykens and Vandewalle, J. Training Multilayer Perceptron Classifiers Based on a Modified Support Vector Method. IEEE Transactions on Neural Networks. 10(4), pp. 907-911, 1999.

[26] Y. Tang. "Deep Learning Using Linear Support Vector Machines," in International Conference on Machine Learning 2013: Challenges in Representation Learning Workshop. Georgia, USA, pp. 2013.

[27] A. F. Agarap. An Architecture Combining Convolutional Neural Network (Cnn) and Support Vector Machine (Svm) for Image Classification. arXiv preprint arXiv:1712.03541. pp., 2017.

[28] J. Nagi, et al. "Convolutional Neural Support Vector Machines: Hybrid Visual Pattern Classifiers for Multi-Robot Systems," in 2012 11th International Conference on Machine Learning and Applications. IEEE. pp. 27-32, 2012.

[29] A. F. M. Agarap. "A Neural Network Architecture Combining Gated Recurrent Unit (Gru) and Support Vector Machine (Svm) for Intrusion Detection in Network Traffic Data," in Proceedings of the 2018 10th International Conference on Machine Learning and Computing. ACM. pp. 26-30, 2018.

[30] F.-J. Huang and LeCun, Y. "Large-Scale Learning with Svm and Convolutional Nets for Generic Object Categorization," in Proceeding. Computer Vision and Pattern Recognition Conference (CVPR'06). pp. 2006.

[31] A. Alalshekmubarak and Smith, L. S. "A Novel Approach Combining Recurrent Neural Network and Support Vector Machines for Time Series Classification," in Proceeding 9th International Conference on Innovations in Information Technology (IIT). IEEE. pp. 42-47, 2013.

[32] M. S. Akhtar, et al. "A Hybrid Deep Learning Architecture for Sentiment Analysis," in Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers. pp. 482-493, 2016.

[33] M. Nabil, Aly, M., and Atiya, A. Labr: A Large Scale Arabic Sentiment Analysis Benchmark. arXiv preprint arXiv:1411.6718. pp., 2014.

[34] A. Y. Al-Obaidi and Samawi, V. W. "Opinion Mining: Analysis of Comments Written in Arabic Colloquial," in Proceedings of the World Congress on Engineering and Computer Science. pp. 2016.

[35] O. Al-Harbi. Using Objective Words in the Reviews to Improve the Colloquial Arabic Sentiment Analysis. International Journal on Natural Language Computing. 6(3), pp. 1-14, 2017.

[36] T. Khalil, et al. "Which Configuration Works Best? An Experimental Study on Supervised Arabic Twitter Sentiment Analysis," in the First International Conference on Arabic Computational Linguistics. IEEE. Cairo, pp. 86-93, 2015.

[37] A. Aliane, et al. "A Genetic Algorithm Feature Selection Based Approach for Arabic Sentiment Classification," in the 13th International Conference of Computer Systems and Applications. IEEE. Agadir, pp. 1-6, 2016.

[38] I. Guellil and Azouaou, F. "Arabic Dialect Identification with an Unsupervised Learning (Based on a Lexicon). Application Case: Algerian Dialect," in 2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES). IEEE. pp. 724-731, 2016.

[39] S. Tartir and Abdul-Nabi, I. Semantic Sentiment Analysis in Arabic Social Media. Journal of King Saud University-Computer and Information Sciences. 29(2), pp. 229-233, 2017.

[40] A. Shoukry and Rafea, A. "A Hybrid Approach for Sentiment Classification of Egyptian Dialect Tweets," in the First International Conference on Arabic Computational Linguistics. IEEE. Cairo, pp. 78-85, 2015.

[41] S. Zhou, Chen, Q., and Wang, X. "Active Deep Networks for Semi-Supervised Sentiment Classification," in Proceedings of the 23rd International Conference on Computational Linguistics: Posters. Association for Computational Linguistics. pp. 1515-1523, 2010.

[42] X. Glorot, Bordes, A., and Bengio, Y. "Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach," in Proceedings of the 28th international conference on machine learning (ICML-11). pp. 513-520, 2011.

[43] C. Dos Santos and Gatti, M. "Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts," in Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers. pp. 69-78, 2014.

[44] D. Tang, Qin, B., and Liu, T. "Aspect Level Sentiment Classification with Deep Memory Network," in Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics. Austin, Texa, pp. 214–224, 2016.

[45] X. Wang, Jiang, W., and Luo, Z. "Combination of Convolutional and Recurrent Neural Network for Sentiment Analysis of Short Texts," in Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers. pp. 2428-2437, 2016.

[46] M. Heikal, Torki, M., and El-Makky, N. Sentiment Analysis of Arabic Tweets Using Deep Learning. Procedia Computer Science. 142, pp. 114-122, 2018.

[47] A. B. Soliman, Eissa, K., and El-Beltagy, S. R. Aravec: A Set of Arabic Word Embedding Models for Use in Arabic Nlp. Procedia Computer Science. 117, pp. 256-265, 2017.

[48] M. Nabil, Aly, M., and Atiya, A. "Astd: Arabic Sentiment Tweets Dataset," in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 2515-2519, 2015.

[49] M. Abdullah, Hadzikadicy, M., and Shaikhz, S. "Sedat: Sentiment and Emotion Detection in Arabic Text Using Cnn-Lstm Deep Learning," in 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE. pp. 835-840, 2018.

[50] R. M. Alahmary, Al-Dossari, H. Z., and Emam, A. Z. "Sentiment Analysis of Saudi Dialect Using Deep Learning Techniques," in 2019 International Conference on Electronics, Information, and Communication (ICEIC). IEEE. pp. 1-6, 2019.

[51] A. A. Altowayan and Tao, L. "Word Embeddings for Arabic Sentiment Analysis," in 2016 IEEE International Conference on Big Data (Big Data). IEEE. pp. 3820-3825, 2016.

[52] A. Barhoumi, et al. "An Empirical Evaluation of Arabic-Specific Embeddings for Sentiment Analysis," in International Conference on Arabic Language Processing. Springer. pp. 34-48, 2019.

[53] N. Abdelhade, Soliman, T. H. A., and Ibrahim, H. M. "Detecting Twitter Users' Opinions of Arabic Comments During Various Time Episodes Via Deep Neural Network," in International Conference on Advanced Intelligent Systems and Informatics. Springer. pp. 232-246, 2017.

[54] A. Al Sallab, et al. "Deep Learning Models for Sentiment Analysis in Arabic," in Proceedings of the second workshop on Arabic natural language processing. Association for Computational Linguistics. Beijing, China, pp. 9-17, 2015.

[55] M. Al-Smadi, et al. Using Long Short-Term Memory Deep Neural Networks for Aspect-Based Sentiment Analysis of Arabic Reviews. International Journal of Machine Learning and Cybernetics. 10(8), pp. 2163-2175, 2019.

[56] M. Al-Smadi, et al. Deep Recurrent Neural Network Vs. Support Vector Machine for Aspect-Based Sentiment Analysis of Arabic Hotels' Reviews. Journal of computational science. 27, pp. 386-393, 2018.

[57] J. Brownlee. Deep Learning for Natural Language Processing: Develop Deep Learning Models for Your Natural Language Problems Machine Learning Mastery. 2017,

[58] R. Collobert, et al. Natural Language Processing (Almost) from Scratch. Journal of machine learning research. 12(Aug), pp. 2493-2537, 2011.

[59] S. Hochreiter and Schmidhuber, J. Long Short-Term Memory. Neural computation. 9(8), pp. 1735-1780, 1997.

[60] M. Sundermeyer, Schlüter, R., and Ney, H. "Lstm Neural Networks for Language Modeling," in Thirteenth annual conference of the international speech communication association. International Speech Communication Association (ISCA). Portland, Oregon, pp. 2012.

[61] M. Schuster and Paliwal, K. K. Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing. 45(11), pp. 2673-2681, 1997.

[62] A. Graves, Jaitly, N., and Mohamed, A.-r. "Hybrid Speech Recognition with Deep Bidirectional Lstm," in 2013 IEEE workshop on automatic speech recognition and understanding. IEEE. pp. 273-278, 2013.

[63] C. Cortes and Vapnik, V. Support-Vector Networks. Machine Leaming. 20(3), pp. 273-297, 1995.

[64] K. A. Kwaik, et al. "Lstm-Cnn Deep Learning Model for Sentiment Analysis of Dialectal Arabic," in International Conference on Arabic Language Processing. Springer. pp. 108-121, 2019.