RESEARCH

Open Access

Stacked-CNN-BiLSTM-COVID: an effective stacked ensemble deep learning framework for sentiment analysis of Arabic COVID-19 tweets



Naglaa Abdelhady^{1*}, Taysir Hassan A. Soliman¹ and Mohammed F. Farghally¹

Abstract

Social networks are popular for advertising, idea sharing, and opinion formation. Due to COVID-19, coronavirus information disseminated on social media affects people's lives directly. Individuals sometimes managed it well, but it often hampered daily activities. As a result, analyzing people's feelings is important. Sentiment analysis identifies opinions or sentiments from text. In this paper, we present an effective model that leverages the benefits of Convolutional Neural Network (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) to categorize Arabic tweets using a stacked ensemble learning model. First, the tweets are represented as vectors using a word embedding model, then the text feature is extracted by CNN, and finally the context information of the text is acquired by BiLSTM. Aravec, FastText, and ArWordVec are employed separately to assess the impact of the word embedding on the our model. We also compare the proposed method to various deep learning models: CNN, LSTM, and BiLSTM. Experiments are performed on three different Arabic datasets related to COVID-19 and vaccines. Empirical findings show that the proposed model outperformed the other models' results by achieving F-measures of 76.76%, 87.%, and 80.5% on the SenWave, AraCOVID19-SSD, and ArCovidVac datasets, respectively.

Keywords Arabic sentiment analysis, BiLSTM, CNN, COVID-19, Ensemble Learning, LSTM, RNN

Introduction

Due to the proliferation of coronavirus, the globe has been in an extremely terrible situation. The perpetual COVID-19 pandemic is one of the main crises of the twenty-first century. This pandemic has had an extraordinary effect on people, both implicitly and explicitly. People are totally reliant on the Internet for activities, such as working from home, and everyone views the coronavirus-related content that flows on social media. Individuals can express their opinions and reviews broadly

Naglaa Abdelhady

and Information, University of Assiut, Assiut, Egypt

through social media sites, like Facebook, Twitter, and Instagram, as well as blogs, review websites, and news websites. Twitter is one of the most effective platforms for understanding user behavior. Twitter has 229 million active users who post publicly every day¹. Analyzing tweets during and after the coronavirus pandemic might be worthwhile because the condition and people's emotions change at every moment throughout this vital period. The reason for this study is to look at how people's feelings and worries about COVID-19 have changed from the start of the pandemic to the present day.

At the present moment, sentiment analysis is regarded as one of the most rapidly developing fields of research

¹ https://www.omnicoreagency.com/twitter-statistics/.



© The Author(s) 2024. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

^{*}Correspondence:

naglaaelhady@aun.edu.eg

¹ Department of Information Systems, Faculty of Computers

inspired by social media platforms. Sentiment analysis is the method of analyzing emotions, opinions, evaluations, and attitudes [1]. Numerous researchers have produced high-quality work in the task of sentiment analysis for English. On the other hand, the amount of effort devoted to sentiment analysis for Arabic is quite restricted because of the intricacy of the Arabic language's orthography and morphology. Arabic has more dialects than English, thus increasing the difficulty and complexity of sentiment analysis, especially when working with noisy, unstructured data from social media. Furthermore, there has been a rise in the quantity of Arabic content on the internet, particularly on social networking platforms. Additionally, the availability of accurate preprocessing tools for Arabic is another current limitation, along with the limited research available in this area. As a result, manually extracting information from the vast amount of online data is a time-consuming and costly task. It also obstructs the process of making the right decision. Therefore, our research focuses on improving the accuracy of the algorithm for classifying Arabic opinions and reducing the human effort required to analyze the text. Several approaches have been developed to handle sentiment analysis and opinion mining challenges in Arabic, such machine learning, lexicon, and hybrid approach. Deep learning approaches have been demonstrated to be superior to other machine learning techniques in terms of achieving greater performance in the sentiment analysis task [2]. Single classifiers have shortcomings and are incapable of achieving superior classification results [3]. As a way to improve results and effectively improve accuracy, ensemble learning has been used extensively in a variety of fields [4–6]. Ensemble techniques combine two different or more of the approaches that are available. Compared to stand-alone techniques, ensemble approaches give better generalization and improved performance, such as bagging, boosting, stacking, etc. [3]. In this study, we are concerned with Arabic sentiment analysis at the sentence level. Specifically, we intend to categorize tweets about COVID-19 in order to identify people's stances on this issue and evaluate the polarity of the tweets, determining whether they are positive, neutral, or negative. Recurrent Neural Network (RNN) is a sort of deep learning model that is efficient at learning sequential models but incapable of extracting local features in a parallel manner. RNN becomes a method that complements CNN due to its ability to maintain information sequences across time. There are several challenges with RNN, including vanishing problems and gradient exploding [7]. Due to these issues, training RNNs for long-distance correlation in a series is challenging. BiLSTM is an RNN model that incorporates two LSTM directions to enhance the network's contexts.

BiLSTM has both backward and forward hidden layers, which gives the network access to the context of the previous and next sequences [8]. On the other hand, text is often represented as vectors in a high-dimensional space. When BiLSTM pulls contextual data from the features, it is unable to prioritize the most important information. Unlike BiLSTM, CNN employs a convolutional layer to extract vector features and minimize their dimension. In the current work, the proposed model solves the aforementioned issue and takes advantage of both models by proposing an ensemble deep learning model for Arabic text classification utilizing CNN and three BiLSTM. The model is trained and assessed using three datasets regarding COVID-19: SenWave, AraCOVID19-SSD, and ArCovidVac datasets. For word embedding, the Aravec, FastText, and ArWordVec embedding models were utilized. The proposed model is evaluated in terms of recall, precision, F-measure, and Area Under the Curve Receiver Operating Characteristics (AUC-ROC). The experimental outcomes demonstrated that our suggested model outperformed competing models.

A summary of the paper's key contributions is as follows:

- Analyze the performance of multiple deep learning models for Arabic sentiment analysis based on distinct network architectures, such as CNN, LSTM, and BiLSTM.
- Develop an effective ensemble deep learning architecture that combines CNN and stacked-BiLSTM to improve the performance of Arabic classification.
- Study the influence of static word embeddings, such as AraVec, FastText, and ArWordVec, on the proposed model.
- Carry out hyperparameter tuning in order to determine which parameters should be used in order to achieve optimal performance.
- Conduct extensive experiments utilizing three public benchmark datasets for Arabic text classification tasks in the context of COVID-19 to verify that the presented architecture significantly surpasses the individual deep learning models.

The remainder of the paper is structured as follows: Related works section offers an overview of recent research on Arabic sentiment analysis based on deep learning and ensemble models; Research methodology section explains the specifics of the proposed model architecture, including methods used to clean the text in the dataset, the word embedding to represent the dataset, and several deep learning models for classification with implementation details. Experimental setup section goes into detail about the experiments and how they turned out. Conclusion and future work section concludes and summarizes the paper and suggests future research.

Related works

This section discusses the current research on Arabic sentiment analysis approaches with an emphasis on deep learning and ensemble methods. A summary of previous research is illustrated in Table 1.

Deep learning approaches in Arabic sentiment analysis

Deep learning is an attractive alternative to traditional machine learning techniques. It has demonstrated outstanding performance on huge datasets across several NLP tasks, including sentiment classification. For example, a study in [9] used Gulf, Iraqi, Egyptian, and Levantine dialect sentences to study four deep learning networks for predicting sentiment polarity, including LSTM; CNN; BiLSTM; and (CLSTM). A subset of the Arabic Online Commentary (AOC) [18] dataset was used to evaluate the four deep learning classifiers. The LSTM achieved the highest overall accuracy result out of all other classifiers, scoring 71.4% across the board for all three of the selected dialects combined.Deep Neural Network (DNN) was applied to perform Arabic sentiment analysis in [10]. The authors' classification model for Arabic tweets consists of eight layers. The sentiment of each tweet is determined by utilizing a lexicon to take the sentiment words out from the tweet and then totaling their polarity. They gathered data from Twitter regarding sports and the Egyptian stock exchange. The testing results indicate that DNN has values of 90.22% accuracy, 90.9% recall, 90.5% precision, and 90.6% F-measure. Arabic microblog sentiment analysis was studied by [5] investigated diverse CNN and LSTM techniques for utilizing six models: CNN, LSTM, three-stacked LSTM layers, CNN with LSTM, and two LSTMs combined with summation, concatenation, and multiplication. Four

Table 1 A review of related works

evaluation metrics were used to evaluate these approaches: precision, recall, accuracy, and F1 score. Arabic Sentiment Tweets Dataset (ASTD) [19] and ArTwitter [20] were utilized as benchmark Arabic tweet datasets. With both a static and a dynamic initialization for Skip-gram (SG) and Continuous Bag-of-Words (CBOW) word embedding, The analyzed models utilized Word2vec as an input. The LSTM performed far better than the CNN in the experiments that were conducted. It was also found that dynamic models with coupled LSTM architectures performed better than the other models, especially when two LSTMs were merged with concatenation for the ArTwitter dataset and with SG word embeddings. he scores were 87.36% for precision, 87.27% for recall, 87.28% for F-measure, and 87.27% for accuracy. On the other hand, the same architecture used for the ArTwitter dataset with CBOW embedding got an accu-F-measure, recall, precision racy, and were 86.45%,86.45%,86.45%, and 86.46% respectively. [11] examined learning algorithms for Arabic sentiment analysis by combining CNN and LSTM, and contrasting them with another combination of BiLSTM and LSTM. The authors utilized three Arabic datasets: ASTD, Shami-Senti, and the Large-Scale Arabic Book Reviews Dataset (LABR) [21], which had varying sizes and dialects. Using Shami-Senti, the model's binary classification accuracy was 93.5% and its three-way classification accuracy was 76.4%. When it came to ASTD, the accuracy reached 85.58% for binary classification and 68.62% for three-way classification. Soufan [22] conducted both binary class and multiclass sentiment classification tasks for the Arabic language using five different methods, including support vector machine, multinomial Naive Bayes, word-level CNNs, LSTM, and char-level CNN models. Five binary datasets were used for the binary class classification task (ArTwitter, QRCI, ASTD, LABR, and Comb.). SemEval-2017 Task4 Sub-TaskA was what they worked on for

Reference	Model	Dataset Arabic Online Commentary		
[9]	LSTM; CNN; BiLSTM; and (CLSTM)			
[10]	DNN	Sports and the Egyptian stock exchange		
[5]	CNN and LSTM	ASTD and ArTwitter		
[11]	CNN, LSTM, and BiLSTM	ASTD, Shami-Senti, and LABR		
[12]	LSTM	twelve Arab nations in four regions		
[13]	CNN	Algerian comments		
[14]	LSTM	AraSenTi		
[15]	LSTM, CNN, and Recurrent CNN	Arabic Twitter corpus		
[4]	CNN, LSTM, and ensemble model	ASTD		
[16]	Ensemble	Syria Tweets		
[17]	Ensemble	SemEval 2018		

the multiclass classification task. For SemEval, the CNN-Word architecture got 50.1% accuracy, which was the highest of any model in the SemEval task. The machine learning methods are more accurate than the other methods when it comes to binary classification. A deep learning framework proposed by [12] identified the polarity of tweets in a 5-scale classification that spans from extremely positive to extremely negative. They collected a total of about 470 thousand tweets from twelve Arab nations in four regions (North Africa, Egypt, the Levant, and the Arab Gulf). The proposed deep learning model architecture is made up of an embedding layer and an LSTM layer. The skip-gram model from a pre-trained Word2Vec is used as word embeddings. The authors test how well their model works on different morphological forms (stem and lemma). For Egyptian tweets and lemmatizations, they received 70% accuracy, while for UAE tweets, they retrieved an accuracy of 63.7%. A framework for analyzing how Algerians feel about comments on the Facebook pages of different Algerian brands is proposed by [13]. The authors collected 100 thousand Algerian comments, but they only labeled 25 thousand of them as positive, negative, or neutral. Feature extraction and transformation are both accomplished using a CNN. To identify the sentiment of a comment, their model is composed of three types of layers: three CNN layers, which each have three kernel sizes and fifty filters, pooling layers, and fully connected layers. An 89.5% accuracy rate is achieved by the model they have developed. According to [14], LSTM was used to explore the impact of several pre-trained word embedding approaches on the model's accuracy by classifying Arabic texts using FastText, ArabicNews, and AraVec as word embedding. They evaluated the proposed framework for three classes: positive, neutral, and negative, using an AraSenTi dataset from [23]. At first, the model did some preprocessing steps on the AraSenTi tweet datasets. After that, vectors for each word in the tweets were constructed using one of the three pre-trained word embeddings. In order to identify if a tweet was positive, neutral, or negative, the embedding was passed to an LSTM layer with a 128-dimensional hidden state. Compared to ArabicNews (91%) and AraVec (88%), FastText has the highest accuracy (93.5%). While the dataset is split evenly into three classes, the low F1 score (40% for AraVec, 43% for ArabicNews, and 41% for Fast-Text) indicates both poor precision and poor recall. Another work was proposed in [15] to categorize tweets in the Arabic Twitter corpus [24]. They applied three different deep learning models: LSTM, CNN, and Recurrent CNN. The results of the experiments show that LSTM performs better than CNN and Recurrent CNN, with an average accuracy of 81.31%. LSTM accuracy is improved by 8.3% when data augmentation is used on the corpus. Study [25] used LSTM, Bi-LSTM, Gated Recurrent Unit (GRU), and Bidirectional-GRU, with different modes (summation, concatenation, average of outputs, and multiplication) for emoji-based tweets to detect sentiment polarity and compared these models with deep neural networks and baseline machine learning classifiers, such as Stochastic Gradient Descent (SGD), Gaussian naive Bayes, Support Vector Machine (SVM), K-nearest neighbor, and decision tree classifiers. These models used a set of 843 Arabic microblogs with emojis from different resources, such as the ASTD, ArTwitter, QCRI [26], Syria, and SemEval-2017 Task 4 Subtask A [27]; in addition, data were scraped from YouTube and Twitter and were manually annotated. They identified emojis in the data set using the Emoji Sentiment Ranking lexicon. The results showed that the GRU and LSTM models outperformed the other models significantly. In particular, the bidirectional GRU outperformed the Bi-LSTM, with an accuracy of 78.71% and an F1 score of 78.76%. Moreover, [28] proposed a method to build a deep learning model for multilabel emotion classification in Arabic tweets using the SemEval2018 Task1 dataset. The model implemented a novel multilayer bidirectional BiLSTM trained on top of pre-trained word embedding vectors. The proposed method achieved the best performance results compared with SVM, random forest, and a fully connected deep neural network. It achieved 9% increase in validation than the previously best obtained by SVM. Abdullah and Shaikh [29] performed punctuation treatment, white space removal, and tokenization. Then, word2vec embedding AraVec was combined with Affective Tweets Weka-package features. At last, the classification is carried out using a fully connected neural network with three dense hidden layers and an SGD optimizer. The model's accuracy was 0.446, which was higher than the baseline model's accuracy.

Ensemble methods in Arabic sentiment analysis

The recent growth in deep learning has also presented existing ensemble learning methods on deep learning classification techniques. Ensemble approaches have been demonstrated to be successful in enhancing results in several domains compared to deep learning's baseline approaches. For example, in [4], researchers have developed two different types of deep learning models: CNN and LSTM. The CNN model is made up of three CNN layers. Each of these layers gets the same word embedding as its input. The results of these layers are added together and sent to a fully connected layer, then to a dropout layer, and finally to a SoftMax function, which is specially designed for multi-class classification tasks.

On the other hand, the LSTM model is built from a bidirectional LSTM. The final output of each layer is added together and sent to a fully connected layer, then a dropout layer, and finally a SoftMax function will figure out what class the input text belongs to. Based on these two models, the authors created an ensemble model that uses soft voting to determine the sentiment class of a text using the Arabic benchmark dataset (ASTD). The ensemble model was correct 65.05% of the time. The LSTM's accuracy was 64.75%, whereas the CNN's accuracy was 64.30%. The ensemble model performed better than other models, with 65.05%. In another study [16] developed several ensemble learning methods using an unbalanced dialectal dataset called the Syria Tweets dataset [30]. The dataset contains 448 positive and 1350 negative tweets, which was obtained in May 2014 by querying the Twitter API for Syria tweets. They tested several ensemble combinations using a pre-trained word2vec word embedding with the CBOW model as a feature. To fix the problem of imbalanced classes, they employ the synthetic minority oversampling technique, which is a way to oversample the minority class by including synthetic instances. Their findings indicate that using word embedding and SMOTE with an ensemble can result in an average F1 score improvement of more than 15% compared with the baseline models. Moreover, [17] developed an ensemble model that incorporated gradient-boosted trees, BiLSTM (word and char level), and CNN models based on different pre-trained embedding and three different Arabic lexicons. The SemEval 2018 dataset was used to categorize four emotions: sadness, fear, joy, and anger. The proposed model got the following results: valance regression 81.6%, valance ordinal classification 75.2%, emotion intensity 68.5%, and emotion intensity ordinal classification 58.7%. In another study presented by [6], the authors retrieved semantic features from short Arabic text at the character level and at the word level. Second, they developed three deep learning models for classification tasks: LSTM, CNN, and an ensemble model that took the best parts of both models to enhance the performance of prediction. To further enhance the neural network's performance, they applied a method called hyperparameter tuning estimation. They used a Twitter dataset of dialectal Arabic and a Modern Standard Arabic (MSA) corpus to train and test the proposed models. This study's results demonstrated remarkable progress in Arabic text categorization, with an accuracy rate ranging from 88% to 69.7%. On the test set, the ensemble model achieved the best accuracy (96.7%). A voting-based ensemble technique, Deep learning for Arabic Sentiment Analysis (DeepASA), has been proposed in a recent study [31] to handle the Arabic sentiment analysis task, which is built on two different types of recurrent network: LSTM and GRU. The basic structure of the model may be broken down into two distinct components. In the first component, text documents, which are represented by FastText realvalued vectors, are used as inputs to LSTM and GRU classifiers to create high-level features. Following that, the output of both classifiers is input into the votingbased ensemble technique, which is majority voting; it comprises of three machine learning algorithms to predict the class for each given text. There are a variety of Arabic datasets used to evaluate the DeepAS's performance, including: ASTD, ArTwitter, Product Reviews (PROD) [32], Restaurant Reviews (RES) [32], Hotel Reviews (HTL) [32], and LABR. With regards to overall performance on all of the chosen datasets, DeepASA was shown to be superior to other methods. In point of fact, DeepASA was able to attain a classification accuracy of 94.32%, and it was also able to minimize the rate of classification errors by as much as 26%.

Research methodology

In this section, we will discuss the steps needed to carry out a sentiment analysis task on Arabic tweets in detail relating to COVID-19, as well as some of the implementation specifics. Figure 1 depicts a general architecture of the Arabic framework for sentiment analysis. As seen in the figure, the sentiment analysis task is comprised of four primary processes: 1) Text Preprocessing, 2) splitting the dataset into training and testing datasets, 3) Selecting the appropriate word embedding approach to represent the dataset, and 4) Experimenting with different deep learning classifier models for identifying the sentiments represented in the original Arabic tweets.

Text preprocessing

It is common for datasets to have inconsistencies, irrelevant data, and duplicate information. Therefore, to achieve a good input for sentiment analysis and a good output for subsequent processing, preprocessing is an important step. Preprocessing of Arabic tweets consists of many phases to decrease word ambiguity, hence improving the precision and efficacy of our proposed model, as shown in Fig. 2 and described below.

Tokenization: is a text-splitting technique that uses blanks (white space, commas, semicolons, periods, and quotes) to separate words from one another. Each of these tokens might be a single word, such as a noun or a verb, or even a preposition or punctuation mark, that is transformed without regard to its meaning or context. The token list serves as an input for subsequent processing.

Cleaning: enhances the effectiveness of the Arabic sentiment analysis task by getting rid of all the Latin characters,

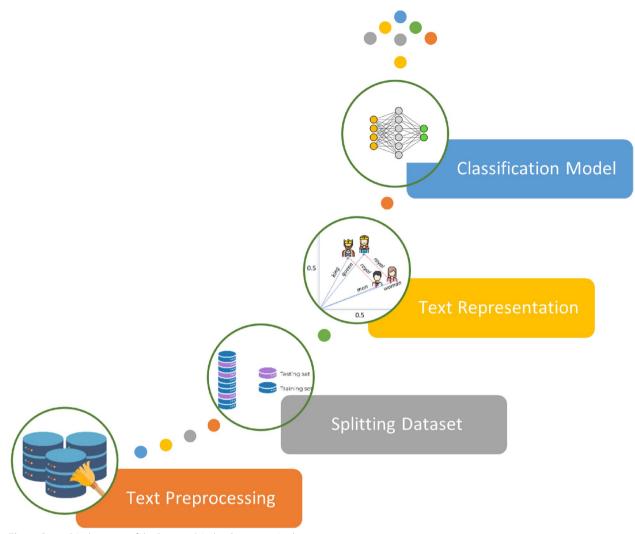


Fig. 1 General Architecture of the Proposed Arabic Sentiment Analysis

URL links, digits, tags, punctuation like (.: "; ;) usernames, images, and special characters like (#?% &') that are in the text of tweet but do not mean anything to learning models and make the feature space more complicated, therefore getting rid of them tends to minimize the feature space.

Removing Duplicate: Due to the fact that Twitter API typically provides identical tweets that contain the same content; hence, we eliminated all identical tweets by matching each tweet to others and also deleting retweet posts.

Removing Stop Words: Stop word removal entails the eradication of minor words, which exist in sentences but have no bearing on the meaning or significance of the document's content, such as articles, pronouns, conjunctions, and prepositions. For the purpose of carrying out this operation, we make use of a list of 742 Arabic stop words that was compiled by [20]. In addition, we omitted

several special stop-words that were found to have a substantial effect on the overall polarity of tweets.

Removing Arabic Diacritics /tashkeel: Some tweets have diacritics despite not being in MSA. In this work, all diacritics were deleted from a tweet.

Removing Elongation: Elongation occurs when characters of a word are repeated without gaps in between. This occurs frequently on social media platforms that indicate affirmation and emphasis, thus in this situation, we attempt to recognize and substitute repeated characters with the letter itself.

Word Normalization: The goal of normalization is to reduce the number of various ways a character may occur in a single word, as seen in Table 2.

Emojis Translation: Emojis are visual symbols, also known as ideograms, that represent not only face reactions but also ideas and concepts such as happiness and

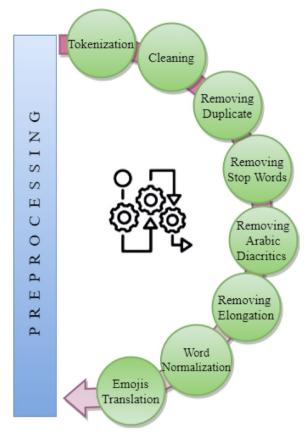


Fig. 2 Preprocessing Steps

 Table 2
 Word normalization

Letter to replace	Replace by		
ĨĮĺ	1		
ö	ھ_		
ي	ى		
ئ ؤ	2		

joy (aughter), sadness (b), disappointed (c), or confuse (c). We might need to transform these emotions into tokens in order to help enhance sentiment analysis performance as follows. Transforming emotions that are compared into predefined lists with words, such as:

- Transform the emotion found in the loves list into the word "حب".

- Transform the emotion found in the smile faces list into the word "ieta - jeta".

- Transform the emotion found in the sad faces list into the word "حزن".

- Transform the emotion found in the neutral faces list into the word "عادى".

Table 3 shows samples of tweets' raw text before and after preprocessing, along with the tweet's polarity.

Splitting dataset

The fundamental principle for constructing a deep learning model is the usage of a training dataset to train the model and a testing dataset to assess its performance. In order to decrease the variance and assure the model's generalizability, data is shuffled prior to data splitting. Additionally, shuffling makes the training set more reflective of the whole data distribution and prevents model overfitting. It was decided that 80% of the dataset would be used for training, and the remaining 20% would be used for testing. During the training phase, the training set will be separated into two sets: 70% of the training set and 30% of the validation set, which will be used to tune the hyperparameters throughout this phase.

Text representation

Word embedding is a way to represent words as numeric vectors. Words that share a common meaning or context are expressed by vectors that are similar in size and shape. Recently, word embeddings are frequently employed to handle a variety of sentiment analysis and NLP issues. Deep learning algorithms use them as input layers. There are several techniques for Arabic word embeddings, such as Word2Vec [33], Doc2Vec [34], Global vectors (GloVe) [35], ArabicNews [36], FastText [37], AraVec [38], and ArWordVec [39]. The words are represented in different ways by each approach. An important role is played by text input, where context can alter the distribution of words. We only investigated three word embedding strategies, namely AraVec, FastText, and ArWordVec, to build text representations in order to verify that our suggested model utilizes the best suitable embedding. Table 4 displays the adjusted hyperparameters for each word embedding model.

AraVec

AraVec developed by [38], is an open-source tool that offers powerful word embedding models for Arabic NLP applications. AraVec includes six pretrained word embedding models that utilize data from Twitter, Wikipedia, and Common Crawl. The total number of tokens utilized to create the models is above 3300000000. For each resource, they've offered two models: one based on the CBOW, while the other is based on the SG model.

Table 3 Examples for tweets before and after preprocessing

Tweet before preprocessing	Tweet after preprocessing	Polarity	
منوتي ليتك معي ـ الله يلعن كورونا و يلعن الصين	منوتي ليتك معي الله يلعن كورونا يلعن الصين حزن	Negative	
ما هى المسافةالتى يجب الحفاظ عليها لتجنب الرذاذ الناتج من العطس و السعال؟ يجب أن تحافظ على مسافةمتر تقريبا بينك وبيني https://t.co/KR84XeMQVo	ما هي المسافه يحب الحفاظ لتجنب الرذاذ الناتج العطس السعال يحب تحافظ مسافه متر تقريبا بينك وبيني	Neutral	
خلاص یا جماعه خلونا ننسی الکورونا و نبدأ کی برسایل رمضان إحنا متأخرین بالمنهج https://t.co/s0Axljm7EL	خلاص يا حماعه خلونا نسي الكورونا نبدا برسايل رمضان احنا متاخرين بالمنهج فرح فرح	Positive	

Table 4 Setting of word embedding

Word Embedding	Setting
AraVec	Word vector dimension = 300, $sg = \{0, 1\}$, min_count = 1, window = 5
fastText	Word vector dimension = 300, $sg = \{0, 1\}$, min_count = 1, window = 5
ArWordVec	Word vector dimension = 300, $sg = \{0, 1\}$, min_count = 1, window = 5

These models were tested using both qualitative and quantitative metrics on a variety of tasks, including the detection of word similarity. The proposed methodology yields considerable results. The method is a good way to figure out how similar two words are and can also improve the performance of other NLP tasks.

FastText

FastText is a word representation developed by Facebook AI research [37] that has been trained on a variety of languages, such as Arabic and English. It is an expansion of the Word2Vec paradigm in which subwords are taken into account. The representation of words as vectors is based on an unsupervised technique. There are two architectural models available: CBOW and SG. Nevertheless, FastText separates each word into n-gram characters. It employs angle brackets as a specific delimiter to indicate the beginning and end of the term. It distinguishes a word from itself and a subword from another word. This model is able to differentiate between prefixes and suffixes, as well as shorter letter sequences, by taking subwords into account. This model consists of the word itself, which is represented as a vector, and its character n-grams. The sub-words are associated with their parent word in a hashtable list, and the total of the n-gram vectors equals the vector of the parent word.

ArWordVec

[39] used 55 million tweets to create ArWordVec, which is a pre-trained word embedding model for the Arabic language. It combines three well-known methods: Glove, Word2Vec CBOW, and Word2Vec Skip-Gram. It is built on Twitter data and has demonstrated superior word similarity scores compared to prior pre-trained algorithms.

Classification models

This section describes the deep learning classification models and how they are utilized in this study. Four distinct model representations are built using CNN, LSTM, BiLSTM, and deep ensemble approaches.

Convolution neural network

CNN [40] is a special form of artificial neural network that is capable of identifying information in various places with high accuracy. This network is used within the field of image processing. However, CNN model has been utilized efficiently in text classification due to its ability to identify local characteristics via the use of convolution kernels and to automatically learn these features for classification solutions. It is distinguished by a special architecture that facilitates learning. In addition, CNN gives an end-to-end learning model whose parameters may be learned using the gradient descent approach.

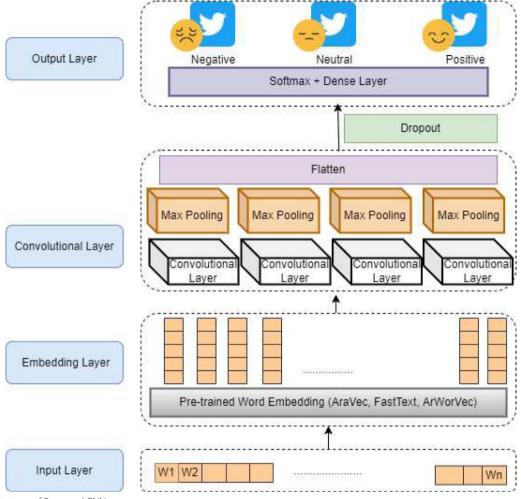


Fig. 3 Architecture of Proposed CNN

As illustrated in Fig. 3, CNN model is composed of four layers: input, convolutional, pooling, and fully connected. First, tweets are transformed into a matrix of numbers, which is then fed into the convolutional layer. Each phrase is composed of words or tokens, and each token corresponds to a row or vector in the matrix table. Typically, these vectors are created using one of three embedding models: AraVec, FastText, or the ArWordVec model. The CNN model accepts vectors as input and extracts local features using filters. The convolutional layer, which is the most significant layer in CNN, does the majority of the feature computations. The convolutional layer generates feature maps using the convolution kernel function. After convolution, the pooling layer captures the most significant features. CNN classifier uses a polling layer to make computing less complicated, where the CNN output size of one stack layer is lowered to the next while preserving fundamental information. This procedure enables the pooling layer to lower feature dimensions, reduces CNN's computing time and cost, and avoids the model from overfitting. There are other polling algorithms available but max-polling is the most common, where the pooled window contains the maximum value element. The output of the polling layer is sent into the flattened layer, which then maps the output to the next layers. Also, dropout is applied between two dense layers to prevent potential overfitting. Our final step is to create a probability distribution for categorizing sentiment scores into three categories: positive, negative, and neutral, using the softmax activation function of the fully connected layer.

Long short term memory

LSTM is one of the most widely known RNN models, having been introduced by [41]. The LSTM model is capable of addressing the issue of vanishing gradients in ordinary RNN and has the ability to capture long term dependencies. This makes them more robust and

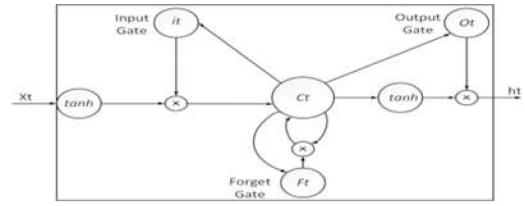


Fig. 4 The Architecture of LSTM Model [42]

adaptable. The LSTM classifier is a variant of RNN that incorporates a mechanism for transporting information over many time steps. Controlling the flow of information in LSTM is accomplished by three distinct gating mechanisms: input, forget, and output gates. Here, the input gate determines how much of the current input is stored in the unit state, the forget gate regulates the selective forgetting of the input at the previous time step, and the output gate controls the output of the current unit state. Figure 4 depicts the operational concept of LSTM.

At a specific time step t, LSTM determines which information must be extracted from the cell's state. The choice is made by a forget gate layer of sigmoid function σ . The function (f_t) accepts the output of the previous hidden layer (h_{t-1}) and the current input (x_t) and returns a value in [0, 1]. In the following equation , the value 1 indicates "reserve all", whereas the value 0 indicates "discard all".

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

where, W^f stands for the forgetting gate's weight matrix. After that, the LSTM decides what new data should be kept in the cell's state. There are two procedures, As seen in Eq. 2, the first step interacts with the "input gate," which is a sigmoid function layer. This function is responsible for specifying an LSTM whose values have been modified. In the second stage, the tanh function layer creates a vector of new candidate values m_t and adds the state of the cell. LSTM combines these steps to initiate the creation of an update to the state.

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

$$m_t = tanh(U_m h_{t-1} + W_m x_t + b_m) \tag{3}$$

where, W_i represents the weight matrix of the input gate; b_i denotes its bias vector;tanh is the hyperbolic tangent function; m_t stands for the new information added to the memory unit; whereas W_m and U_m denote the associated weight matrix and bias.

At this time, the model change the old cell state C_{t-1} into a new cell state C_t as depicted by Eq. 4. It's worth noting that the gradient can be controlled when passing through the forget gate f_t and enables for updates and deletes for explicit "memory". This approach aids in preventing gradients from vanishing or any issues linked with the exploding gradient in the standard RNN.

$$c_t = f_t * c_{t-1} + i_q * m_t \tag{4}$$

Finally, LSTM determines its output depending on the current state of the cell. LSTM permits a sigmoid layer where it decides which portions of the cell state to transfer as output in Eq. 5, referred to the "output gate". At this point, LSTM determines the state of the cell based on the function tanh and decides the output using Eq. 6.

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

$$h_t = o_t * tanh(c_t) \tag{6}$$

Figure 5 shows specifics of the LSTM model design. The model began by producing an input matrix containing 300-dimensional word vectors for each word in a tweet. The word embedding values were loaded using three different pre-trained word embedding vectors, including AraVec, FastText, and ArWordVec. Each word embedding was individually input to the LSTM layer. The embedding was then passed to the LSTM layer, which had a 300-dimensional hidden state. Following that, a dense layer with a Rectified Linear Unit (ReLU) activation function follows. The reasons we chose ReLU are to avoid the problem of vanishing gradients and to speed up the calculation. The returned sequences were subjected to a dropout fraction rate of 0.5. Lastly, a dense thick layer with three units was utilized to assign one of the three possible classes, followed by softmax activation.

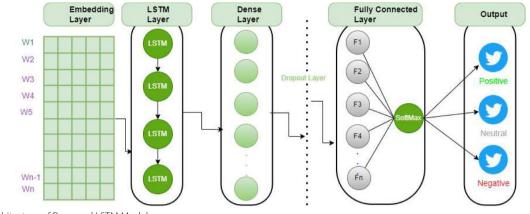


Fig. 5 Architecture of Proposed LSTM Model

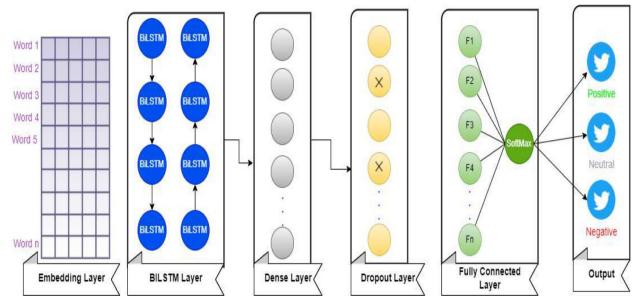


Fig. 6 BiLSTM Proposed Model Architecture

Bidirectional LSTM

BiLSTM is a sort of RNN designed to solve LSTM's deficiencies with text sequence features. Information in LSTM goes from backward to forward, but BiL-STM uses two hidden states to flow information in two directions: backward to forward and forward to backward. BiLSTM is a pioneer in the field of sentiment classification because of its structure, which helps it to learn context more effectively than other models. BiL-STM retains input data from both the prior and following sequences, unlike the typical RNN model, which requires decay in order to include future information.

Figure 6 presents the BiLSTM model's architecture, where this model contains an embedding layer that is

used to transfer word indices into an embedding space. Specifically, given a tweet consisting of n words w_1,w_2 , ..., w_n , we first map each word w_n into a word embedding r_m^w . The sequence of word embeddings with a length of n is passed to the upcoming layer, which is a BiLSTM layer. The model is better able to interpret context since the input information is preserved in the hidden state both from the past to the future (right to left for the Arabic language), and from the future to the past (the opposite sentence direction). The network uses a dense layer as its hidden layer. After this layer comes a dropout layer with a 0.5 rate. Dropout is utilized to regularize and prevent overfitting issues. The Adam optimizer is being used to optimize our model

Page 12 of 21

since it is easy to implement, has a faster execution time, consumes less memory, and requires less tuning than any other optimization strategy. Finally, the output layer, which consists of a dense layer containing three Softmax cells, is then utilized for classification.

Ensemble model

Due to the fact that CNN can only extract local features of text and is insensitive to the order of time, it can not do a good job of figuring out what the text means [3, 7]. On the other hand, BiLSTM improves the contexts available to deep neural networks by using two LSTM directions, but it is unable to extract local features in parallel. Therefore, achieving the best possible classification results for sentiment analysis cannot be accomplished by using a single CNN or single BiLSTM. So, this study presents a hybrid framework that integrates CNN and BiLSTM models.

A stacked ensemble Stacked-CNN-BiLSTM-Covid model uses a convolution layer for extracting the local features from word embeddings and uses the BiLSTM to learn the local features in two two-direction sequences, as illustrated in Fig. 7. Briefly, the procedure began with the embedding layer transforming the input text into a collection of embedding vectors and transferring it to the convolution layer, which extracted the local features and generated feature maps. The CNN layer consists of a one-dimensional convolutional layer in which the filter window size is 3, the number of filters is 300, and the activation function is ReLU. The pooling layer is then used to pool all of the feature maps. Following the pooling layer, each feature map vector is sent via a dropout layer to prevent the neural network from being overfitting. This layer provides a regularization strategy for this deep learning model. In addition, it improves the network's generalization approaches by ensuring that all inputs in the BiL-STM layers are considered without favoring a single one. This layer eliminates any potential biases in the training of these deep learning models. The dropout layer's output is then transmitted to the BiLSTM layer, which uses it to learn the sequences of its input and generate new encoding output.

We stacked three BiLSTM layers on top of CNN to process the input sequence. Similar to BiLSTM, Stacked BiLSTM is capable of getting rich contextual information from past and future time sequences. However, unlike BiLSTM, stacked BiLSTM contains additional upper layers for conducting additional feature extractions, whereas BiLSTM has just one hidden layer for each direction to extract features. After painstaking efforts, we decided to limit the number of LSTM layers in our model to three since we observed that increasing the number of hidden layers needs more processing time with no discernible gain in performance. The last layer of our proposed model is a dense layer comprised of three neurons that classify tweets as positive, neutral, or negative. Softmax is the activation function of the last layer. The softmax function produces a value between 0 and 1 for each target class, indicating its likelihood. We utilized the Adam optimization approach to update the network's weights and a loss function based on categorical-crossentropy.

Regularization

Regularization is controlled by multiple functions that structure a complicated neural network to prevent overfitting, which negatively affects the performance of deep learning models. Different techniques are employed to reduce overfitting in deep learning. Following are descriptions of the three primary strategies we employ in our research: dropout, L2, and early stopping.

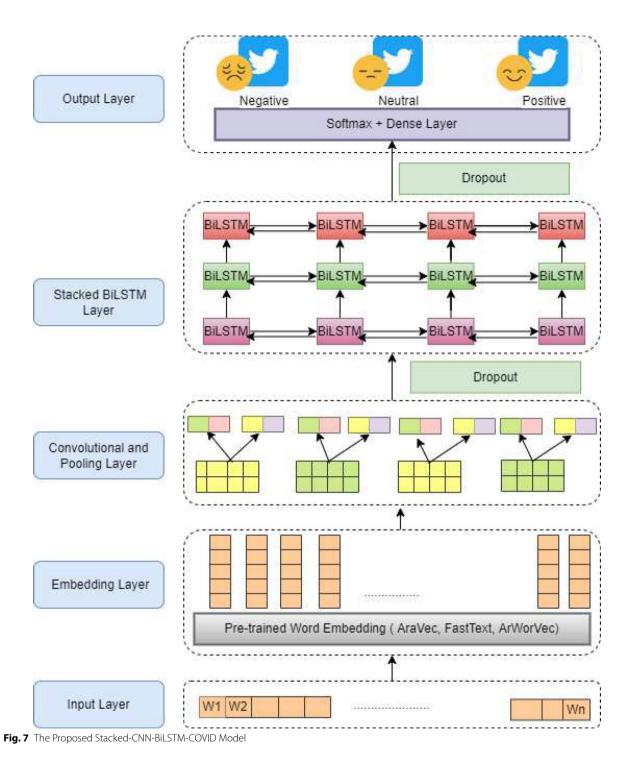
Dropout randomly eliminates a proportion of units at each step of the training phase by setting them to zero. It prevents the model from learning the same values several times in the event that there are a multitude of parameters. The LSTM layer's recurrent dropout regularization value was set at 0.5. Each BiLSTM layer has a dropout rate of 0.5.

L2 regularization alongside class weights is implemented in the loss function to avoid excessively large weights and to care for class imbalance. It adds a regularization component to the cost function, hence decreasing the weight matrix values. The value of the hyperparameter λ in the proposed study is 0.001.

Early Stopping is a strategy that terminates training as soon as the performance of the model on the validation dataset stops getting better, regardless of the number of epochs chosen to avoid overfitting. Ending training in our sentiment analysis system is done by using the callback function. The callbacks record the performance values for each epoch, which include the validation loss, validation accuracy, training loss, and training accuracy. Monitor and patience are two crucial factors in this function. We decided to use training accuracy as a monitor, which implies that it will keep a record of any training data loss. The value of the Patience parameter is set to three to ensure that the training process will be terminated, and the model weights will be frozen in place if the accuracy rate does not increase after three epochs have passed.

Experimental setup

To evaluate the proposed stacked ensemble model for Arabic sentiment analysis, we developed an experiment in which several deep learning methods are compared with varying pre-trained word embeddings. Three datasets are used to evaluate these approaches for sentencelevel classification in the COVID-19 domain.



Development environment

We conducted our experiments using Google services. For the purpose of storing our dataset, we made use of Google Drive, a cloud-based file storage service that is offered by Google. This service enables users to store files on the servers as well as share them with other people. In addition, we utilized the Google Colaboratory platform. This platform is a free cloud service provided by Google for developers and is compatible with Jupyter notebooks. Furthermore, the deep learning framework

Table 5 Hyperparameters settings for implementing the deeplearning models

Parameter	Value
Pool Size	2
Kernal Size	3
Filter	300
Learning Rate	0.0001
Optimizer	Adam
Loss Function	categorical crossen- tropy
Output Activation	Softmax
Batch Size	128
Number of epochs	15
Dropout rate	0.5
Recurrent dropout rate	0.5
Kernel Regularizer	L2 (0.001)

Table 6 Details of the datasets used in this study

	SenWave	AraCOVID19-SSD	ArCovidVac
Number of positive tweets	1562	1762	7968
Number of negative tweets	2750	955	638
Number of neutral tweets	4269	1831	1396
Number of emojis	2589	3714	4499
Number of words	122005	72736	230104
Number of characters	678915	493970	1632464
Average words per tweet	14	14	23

employed was Keras², which was developed on top of Tensorflow. For plotting, the Matplotlib Python package is utilized. The values of the hyper-parameters according to the deep learning models' configurations are shown in Table 5. Typically, the loss function for the multi-classification task is categorical crossentropy. During training, a dropout layer is introduced, and the initial learning rate is set at 0.5 in order to avoid overfitting the model. Optimization is used to update model parameters (weights and bias values) across iterations while training deep learning algorithms. An Adaptive Moment Estimation (Adam) optimizer was utilized in our research. The size of every embedding utilized in experiments was 300. The experiment is repeated 15 times, with 128 being the batch size.

Arabic sentiment benchmark datasets

Unfortunately, the number of publicly available annotated Arabic datasets suitable for use in COVID-19 sentiment analysis is relatively small. In addition, a small number of these datasets are large enough to be utilized in deep learning models. In this research, we only glance at Arabic datasets with three classes because we use a three-way classification model. More information about the statistics of the used datasets can be found in Table 6. We employed the following three Arabic sentiment datasets:

SenWave

SenWave [43] contains 10,000 tweets gathered between 1 March 2020 and 15 May 2020. It is available for download after having been manually annotated. Each tweet was annotated by at least three Arabic experts, and all annotations were submitted for a rigorous quality check.

AraCOVID19-SSD

Ameur and Aliane [44] create and share AraCOVID19-SSD, an annotated Arabic dataset regarding COVID-19. This dataset comprises 5162 tweets that have been annotated for sentiment analysis and identification of sarcasm. The collection of tweets took place between the 15th of December 2019 and the 15th of December 2020. All tweets within the dataset were manually annotated and verified by humans. The authors only provided us with the tweet IDs, so we had to construct a Python script to access the text of the tweets. Due to the absence of data for 614 tweet-ids, the total number of tweets in the AraCOVID19-SSD dataset is 4548 after hydration.

ArCovidVac

ArCovidVac [45] is a manually labeled Arabic Twitter dataset for the COVID-19 vaccine campaign, including numerous Arab nations. defining their opinion on vaccination and the immunization procedure. The authors categorize them as positive (in favor of vaccination), negative (opposed to vaccination), or neutral. The collected tweets were in the period between January 5 and February 3, 2021. To collect these tweets that specified the Arabic language, they made use of the twarc³ search API. In all, they collected 550 thousand unique tweets. Among them, 10 thousand tweets were selected at random for manual annotation.

² https://keras.io.

³ https://github.com/DocNow/twarc

DL Model	Word Embedding	Model	Precision	Recall	F-Measure	ROC
CNN	AraVec	CBOW	74.14%	73.74%	73.93%	87.56%
		SG	74.32%	74.04%	74.17%	87.89%
	FastText	CBOW	70.98%	68.78%	69.6%	84.55%
		SG	68.58%	67.95%	67.69%	82.79%
	ArWordVec	CBOW	72.24%	72.21%	71.85%	87.38%
		SG	74.19%	72.62%	71.64%	87.26%
LSTM	AraVec	CBOW	76.12%	75.52%	75.47%	89.29%
		SG	76.25%	75.81%	75.79%	89.97%
	FastText	CBOW	73.4%	70.9%	70.84%	85.13%
		SG	72.9%	70.61%	70.69%	84.78%
	ArWordVec	CBOW	76.82%	76.11%	76.13%	89.64%
		SG	76.08%	76.17%	76.07%	89.64%
Bilstm	AraVec	CBOW	76.14%	75.28%	75.42%	89.56%
		SG	75.81%	75.52%	75.44%	89.84%
	FastText	CBOW	72.21%	62.63%	65.07%	80.97%
		SG	69.11%	69.01%	69.06%	84.07%
	ArWordVec	CBOW	77.23%	76.35%	76.38%	90.04%
		SG	76.78%	75.93%	75.96%	89.55%
Ensemble	AraVec	CBOW	78.18%	75.93%	76.32%	88.81%
		SG	77.07%	75.81%	76.09%	89.7%
	FastText	CBOW	67.16%	62.68%	64.1%	78.94%
		SG	70.56%	61.44%	64.28%	79.12%
	ArWordVec	CBOW	78.48%	76.23%	76.76%	90.43%
		SG	76.19%	75.22%	75.52%	89.22%

Results and discussion

There are four primary measures: precision, recall, F-measure, and ROC that we utilize to judge our performance in this work. In the upcoming subsections, the experiments and their findings are discussed in detail.

Evaluation criteria

This section discusses the evaluation metrics employed in Arabic sentiment analysis. Data balance is an important issue and the metrics utilized must be applicable to the selected task. When there is an imbalance in the data, accuracy is not the appropriate metric since it is biased toward the majority class. Similarly, it cannot be used as a metric to solve multi-class problems. Evaluation metrics such as precision, sensitivity, AUC, f1-score, log loss, Cohen Kappa score, and others must be utilized [46]. So, we employed precision, recall/Sensitivity, F-measure, and AUC-ROC for model evaluation. Precision can be defined as the ratio of correctly predicted positive occurrences to the total number of predicted positive occurrences. Recall estimates the ratio between the number of correctly predicted positive instances and the total of instances that were predicated as positive and negative. The F1 score provides a concise overview of the performance of the model by combining precision and recall.

$$Precision = \frac{TP}{(TP + FP)} \tag{7}$$

$$Recall/Sensitivity = \frac{TP}{(TP + FN)}$$
(8)

$$F - Measure = \frac{(2.Precision.Recall)}{(Precision + Recall)}$$
(9)

True Positive (TP) represents the correct identification of tweets as positive, False Negative (FN) represents the incorrect identification of tweets as negative, and False Positive (FP) represents the incorrect identification of tweets as positive.

The AUC-ROC curve is a performance metric for classification issues with different threshold values. ROC is a probability curve, whereas AUC is a measure or degree of separability. It indicates how well the model can differentiate across classes. A higher AUC value indicates that the classifier is effective at distinguishing unique classes. AUC is beneficial even when class data is imbalanced.

Table 8 A comparison	n evaluation of four DL models with	n different word embedding for AraCOVID19-SSD dataset

DL Model	Word Embedding	Model	Precision	Recall	F-Measure	ROC
CNN	AraVec	CBOW	66.25%	65.79%	64.32%	83.55%
		SG	65.18%	64.75%	58.76%	85.17%
	FastText	CBOW	77.24%	71.53%	63.96%	92.76%
		SG	77.93%	71.53%	64.69%	93.26%
	ArWordVec	CBOW	65.37%	64.75%	62.74%	83.26%
		SG	73.41%	66.82%	59.93%	86.36%
LSTM	AraVec	CBOW	77.85%	76.92%	76.5%	90.82%
		SG	82.47%	82.32%	82.2%	93.91%
	FastText	CBOW	86.32%	86.11%	86.03%	95.05%
		SG	86.93%	86.8%	86.68%	95.9%
	ArWordVec	CBOW	78.54%	78.19%	78.05%	90.31%
		SG	81.16%	81.06%	80.97%	93.18%
Bilstm	AraVec	CBOW	78.89%	78.19%	77.95%	91.87%
		SG	81.93%	81.63%	81.41%	93.78%
	FastText	CBOW	86.99%	87.03%	86.98%	95.54%
		SG	87.16%	86.82%	86.99%	95.95%
	ArWordVec	CBOW	77.96%	77.84%	77.73%	90.91%
		SG	82.76%	82.66%	82.58%	93.57%
Ensemble	AraVec	CBOW	77.79%	77.15%	76.32%	90.08%
		SG	84.08%	83.93%	83.87%	93.62%
	FastText	CBOW	87.25%	87.26%	87.25%	97.2%
		SG	86.68%	86.45%	86.29%	95.63%
	ArWordVec	CBOW	76.01%	75.77%	75.4%	88.17%
		SG	80.98%	80.94%	80.74%	92.39%

Model evaluation

Results of the performance evaluation of several deep learning models and a stacked ensemble model are provided in this section. The performance of our proposed model is evaluated using three distinct datasets, as described in Arabic sentiment benchmark datasets section. Consequently, three experiments were conducted for every model based on the different embeddings. The performance of models is investigated, as well as the effects of various embedding representations. Tables 7, 8, and 9 show the comparative details of the insights gained for the various models. The greatest values attained by models across all embeddings are emphasized in bold font. Table 7 presents the testing recall, precision, F-measure, and ROC performance of four deep learning models for Arabic sentiment analysis using the SenWave dataset. Considering the CNN model, AraVec with SG architecture obtained high precision, recall, F-measure, and ROC (74.32%, 74.04 %, 74.17 %, and 87.89 %, respectively) with an embedding dimension of 300. The lowest F-measure was attained using FastText word embedding with SG, which was 67%. For the LSTM model at dimension 300, AraVec with SG architecture again achieved the greatest precision, recall, F-measure, and ROC (76.25%,

75.81%, 75.79%, and 89.97%, respectively). In contrast, FastText word embedding with SG architecture scored the lowest F-measure at 72.6%. For the BiLSTM model, ArWordVec with CBOW architecture did the best in terms of precision, recall, F-measure, and ROC (77.23%, 76.35%, 76.38%, and 89.84%, respectively, at dimension 300). On the other hand, FastText word embedding with CBOW architecture had the lowest recall rate, at 62.63%. Lastly, for the ensemble model, ArWordVec with CBOW architecture had the best testing F-measure score of 76.76% and the best ROC of 89.43%. It also had the best testing precision of 78.48% and the best testing recall of 76.23%.

Table 8 displays the testing recall, precision, F-measure, and ROC performance of four deep learning models for Arabic sentiment analysis using the AraCOVID19-SSD dataset. Concerning the CNN model, FastText with SG architecture showed superior results in terms of accuracy, recall, F-measure, and ROC (77.93%, 71.53%, 64.69%, and 93.26%, respectively) at an embedding dimension of 300. In contrast, AraVec word embedding with SG attained the lowest level of precision at 58.76%. When it came to the LSTM model, FastText with SG architecture had the highest results in terms of precision,

DL Model	Word Embedding	Model	Precision	Recall	F-Measure	ROC
CNN	AraVec	CBOW	71.68%	78.55%	73.58%	74.01%
		SG	63.52%	79.7%	70.7%	64.53%
	FastText	CBOW	63.52%	79.7%	70.7%	66.02%
		SG	63.52%	79.1%	70.45%	69.49%
	ArWordVec	CBOW	74.47%	79.95%	74.06%	74.29%
		SG	63.52%	79.7%	70.7%	64.56%
LSTM	AraVec	CBOW	78.68%	79.55%	79.1%	84.38%
		SG	78.31%	79.5%	78.65%	83.89%
	FastText	CBOW	78.71%	78.3%	76.85%	82.19%
		SG	76.51%	73.95%	74.84%	79.61%
	ArWordVec	CBOW	78.74%	79.85%	79.29%	84.75%
		SG	77.86%	79.25%	78.32%	82.95%
Bilstm	AraVec	CBOW	77.72%	80.75%	78.74%	83.83%
		SG	78.14%	80.1%	79.10%	84.09%
	FastText	CBOW	63.52%	79.7%	70.7%	70.59%
		SG	78.4%	76.4%	76.48%	82.07%
	ArWordVec	CBOW	77.04%	80.5%	77.77%	83.29%
		SG	79.34%	80.8%	79.36%	84.17%
Ensemble	AraVec	CBOW	75.58%	80.1%	76.61%	83.99%
		SG	79.02%	78.85%	78.85%	84.12%
	FastText	CBOW	76.89%	79.6%	78.22%	82.57%
		SG	77.05%	75.6%	76.22%	80.74%
	ArWordVec	CBOW	80.72%	80.3%	80.5%	85.09%
		SG	79.3%	79.05%	78.51%	83.79%

recall, F-measure, and ROC (with respective values of 86.93%, 86.8%, 86.68%, and 95.9%). In contrast, AraVec word embedding with CBOW architecture achieved the lowest precision of 77.85%, recall of 76.92%, and F-measure of 76.50%. The lowest ROC recorded by ArWordVec was 90.31% using the CBOW architecture. Regarding the BiLSTM model, FastText with SG architecture once again had the best precision, recall, F-measure, and ROC results (87.16%, 86.82%, 86.99%, and 95.9%, respectively). In contrast, ArWordVec word embedding with CBOW architecture yielded the worse performance in terms of precision, recall, F-measure, and ROC (77.96%, 77.84%, 77.73%, 90.91%). For the ensemble model, FastText with CBOW architecture attained the highest testing accuracy of 87.25%, the highest testing recall of 87.26%, the highest testing F-measure score of 87.25%, and the highest ROC of 97.2%. In contrast, AraWordVec word embedding with CBOW architecture had the lowest performance in terms of precision, recall, F-measure, and ROC (76.01%, 75.77%, 75.4%, and 88.17%).

Table 9 presents the testing recall, precision, F-measure, and ROC performance of four deep learning models for Arabic sentiment analysis using the ArCovidVac dataset. Regarding the CNN model, ArWordVec with CBOW architecture exhibited high precision, recall, F-measure, and ROC (74.47%, 79.95%, 74.06%, and 74.29%, respectively) at an embedding dimension of 300. In contrast, FastText word embedding with SG got the lowest recall of 79.1%. When it came to the LSTM model, ArWord-Vec with CBOW architecture once again had the greatest performance in terms of precision, recall, F-measure, and ROC (respectively 78.74%, 79.85%, 79.29%, and 84.75%). On the other hand, FastText word embedding with SG architecture, had the lowest precision (76.51%), recall (73.95%), F-measure (74.84%), and ROC (79.61%). For the BiLSTM model, ArWordVec with the CBOW architecture once again scored the best in terms of precision, recall, F-measure, and ROC (79.34%, 80.8%, 79.36%, and 84.17%, respectively) at dimension 300. However, the lowest precision was reached by FastText word embedding using the CBOW architecture, at 63.52%. For the ensemble model, ArWordVec with the CBOW architecture scored the highest testing precision of 80.72%, the highest testing recall of 80.3%, the highest testing F-measure score of 80.5%, and the highest ROC of 85.09%. Figure 8 represents the superior F-measure for each model based on three pre-trained word embeddings for the three selected datasets. Regarding SenWave,

SenWave Dataset

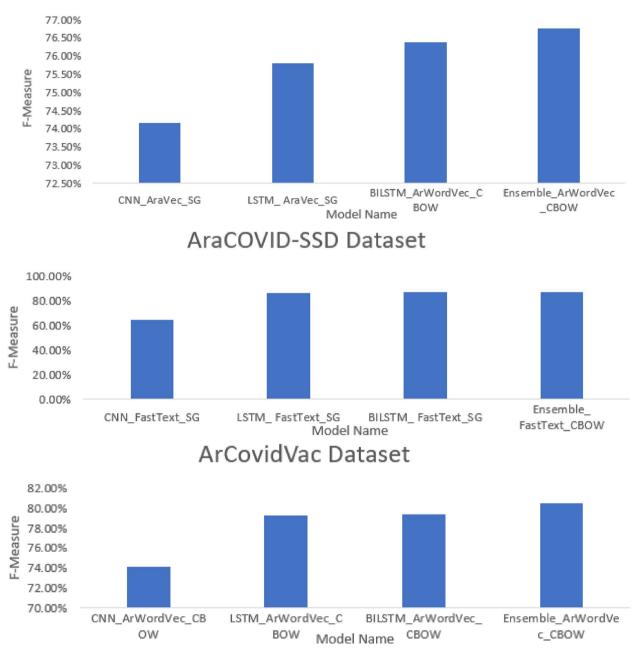


Fig. 8 Best F-Measure for all Models using Different Dataset

AraVec's SG model architecture generated superior performance on CNN and LSTM as compared to other architectures. Meanwhile, ArWordVec with the CBOW model performed better with BiLSTM and the stacked ensemble model. For AraCOVID19-SSD, FastText with the SG model architecture achieved better accuracy than other architectures in almost all models, with the exception of the ensemble model. The ensemble model with FastText (CBOW) achieved 0.96% better results than the ensemble model with FastText (SG). Furthermore, for the ArCovidVac dataset, ArWordVec with the CBOW model worked more effectively on CNN, LSTM, BiLSTM, and

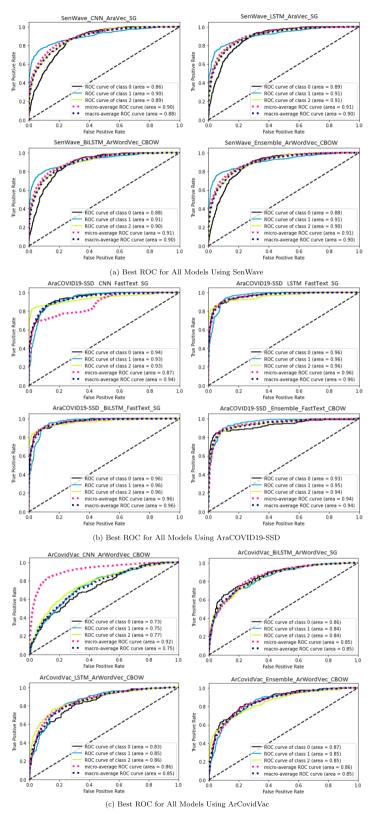


Fig. 9 Best ROC for All Models Using Different Dataset

the proposed stacked ensemble model. This highlighted that word vectorization approaches impact the accuracy of the model.

The proposed stacked ensemble model outperforms existing deep learning models on both the SenWave and AraCOVID19-SSD datasets, as well as the ArCovidVac dataset. Considering SenWave, StackedCNNBiLSTM-Covid performed 2.59% better than the CNN model, 0.97% better than the LSTM model, and 0.38% better than BiLSTM. For the AraCOVID19-SSD dataset, the suggested model's F-measure was 22.56%, 0.57%, and 0.26% higher than CNN, LSTM, and BiLSTM, respectively. Regarding ArCovidVac, the overall F-measure of the proposed model was improved by 6.44% compared to the CNN model, 1.21% compared to the LSTM model, and 1.14% compared to the BiLSTM model. It was noticed that applying sentiment analysis using CNN or BiLSTM alone could not achieve an efficient result since the F-measure of CNN alone was only 74.17% and the F-measure of BiLSTM was 76.38% on the SenWave dataset. Similarly, the F-measures of CNN and BiLSTM on the AraCOVID19-SSD dataset were only 64.69% and 86.99%, respectively. Likewise, the F-measures of CNN and BiLSTM alone on the ArCovidVac dataset were only 74.06% and 79.36%, respectively. This indicates that CNN and BiLSTM cannot produce satisfactory results on their own, as CNN cannot learn the correlation sequence for long-term dependencies and BiLSTM cannot capture local features. When CNN and BiLSTM are combined, the model can learn each word of tweets more effectively since it has sufficient word context information based on the past and future context of the word. Another observation was that BiLSTM outperformed LSTM since it had knowledge of the text's previous and following information.

Figure 9 represents AUC and ROC scores for each deep learning approach employing distinct word embeddings. It is clear that the ROCs of the proposed model and the other models are different. Our classification technique yields an AUC of over 0.64 for all the models. The AUC score for the proposed model ranged between 0.79 and 0.97 and performed consistently better with all word representations. Therefore, it may be concluded that the proposed model accurately scaled maximum sentences.

Conclusion and future work

Using Twitter as a source of data to assess public reaction to epidemic outbreaks has attracted a great deal of attention among researchers due to the rising frequency of pandemic outbreaks. The purpose of this study was to develop a model to get insights regarding the public reaction to COVID-19 based on tweets written in Arabic. We presented a stacked ensemble learning model for Arabic sentiment analysis by combining CNN and stacked BiLSTM. We started by employing the word embedding approach to extract the semantic features of words and convert them into high-dimensional word vectors. We studied the performance of three Arabic pretrained embedding models: AraVec, FastText, and ArWordVec. Next, CNN was used to extract text features, and BiL-STM extracted text context information. The sentiment scores were then categorized as positive, neutral, or negative using a softmax function. In addition, we conducted exhaustive experiments on three benchmark datasets. Experiments proved that the proposed model outperformed competing models by achieving superior F-measures in all datasets. It was found that applying CNN or BiLSTM alone for sentiment analysis could not produce an effective result, and that word embedding approaches affect the model's accuracy. For future research, the structure of the Stacked-CNN-BiLSTM-COVID model could be adjusted to improve sentiment classification performance. Word embedding approaches are another aspect where the model could be enhanced. The experimental findings demonstrated that word representation could impact the overall model's accuracy. As a result, a more comprehensive approach to word embedding could result in improved feature extraction for the network. Additionally, training the models on more large-scale and balanced datasets can improve the quality of the produced models. Additionally, it's worthwhile to look into cutting-edge models like transformers models.

Acknowledgements

Not applicable.

Authors' contributions

Naglaa Abdelhady, Taysir Hassan A. Soliman, and Mohammed F. Farghally contributed to the design and implementation of the research, to the analysis of the results and to the writing of the manuscript.

Funding

Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

Availability of data and materials

Not applicable.

Declarations

Ethics approval and consent to participate

This research did not contain any studies involving animal or human participants, nor did it take place on any private or protected areas.

Competing interests

The authors declare no competing interests.

Received: 26 October 2022 Accepted: 22 March 2024 Published online: 09 April 2024

References

- 1. Liu B (2012) Sentiment analysis and opinion mining. Synth Lect Hum Lang Technol 5(1):1–167
- Collobert R, Weston J, Bottou L, Karlen M, Kavukcuoglu K, Kuksa P (2011) Natural language processing (almost) from scratch. J Mach Learn Res 12(ARTICLE):2493–2537
- Luo S, Gu Y, Yao X, Fan W (2021) Research on text sentiment analysis based on neural network and ensemble learning. Rev d'Intelligence Artif 35(1):63–70
- Heikal M, Torki M, El-Makky N (2018) Sentiment analysis of arabic tweets using deep learning. Procedia Comput Sci 142:114–122
- Al-Azani S, El-Alfy ESM (2017) Hybrid deep learning for sentiment polarity determination of arabic microblogs. In: International Conference on Neural Information Processing. Springer, pp 491–500
- 6. Alwehaibi A, Bikdash M, Albogmi M, Roy K (2021) A study of the performance of embedding methods for arabic short-text sentiment analysis using deep learning approaches. J King Saud Univ-Comput Inf Sci
- Tam S, Said RB, Tanriöver ÖÖ (2021) A convbilstm deep learning model-based approach for twitter sentiment classification. IEEE Access 9:41283–41293
- Liu G, Guo J (2019) Bidirectional lstm with attention mechanism and convolutional layer for text classification. Neurocomputing 337:325–338
- 9. Lulu L, Elnagar A (2018) Automatic arabic dialect classification using deep learning models. Procedia Comput Sci 142:262–269
- Abdelhade N, Soliman THA, Ibrahim HM (2017) 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
- 11. Abu Kwaik K, Saad M, Chatzikyriakidis S, Dobnik S (2019) Lstm-cnn deep learning model for sentiment analysis of dialectal arabic. In: International Conference on Arabic Language Processing. Springer, pp 108–121
- Baly R, El-Khoury G, Moukalled R, Aoun R, Hajj H, Shaban KB, El-Hajj W (2017) Comparative evaluation of sentiment analysis methods across arabic dialects. Procedia Comput Sci 117:266–273
- Soumeur A, Mokdadi M, Guessoum A, Daoud A (2018) Sentiment analysis of users on social networks: Overcoming the challenge of the loose usages of the algerian dialect. Procedia Comput Sci 142:26–37
- Alwehaibi A, Roy K (2018) Comparison of pre-trained word vectors for arabic text classification using deep learning approach. In: 2018 17th IEEE international conference on machine learning and applications (ICMLA). IEEE, pp 1471–1474
- 15. Mohammed A, Kora R (2019) Deep learning approaches for arabic sentiment analysis. Soc Netw Anal Min 9(1):1–12
- Al-Azani S, El-Alfy ESM (2017) Using word embedding and ensemble learning for highly imbalanced data sentiment analysis in short Arabic text. Procedia Comput Sci 109:359–366
- Abdou M, Kulmizev A, i Ametllé JG (2018) Affecthor at semeval-2018 task 1: A cross-linguistic approach to sentiment intensity quantification in tweets. In: Proceedings of The 12th International Workshop on Semantic Evaluation. pp 210–217
- Zaidan O, Callison-Burch C (2011) The arabic online commentary dataset: an annotated dataset of informal arabic with high dialectal content. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp 37–41
- Nabil M, Aly M, Atiya A (2015) Astd: Arabic sentiment tweets dataset. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp 2515–2519
- Abdulla NA, Ahmed NA, Shehab MA, Al-Ayyoub M (2013) Arabic sentiment analysis: Lexicon-based and corpus-based. In: 2013 IEEE Jordan conference on applied electrical engineering and computing technologies (AEECT). IEEE, pp 1–6
- Aly M, Atiya A (2013) Labr: A large scale arabic book reviews dataset. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp 494–498

- 22. Soufan A (2019) Deep learning for sentiment analysis of arabic text. In: Proceedings of the ArabWIC 6th Annual International Conference Research Track. pp 1–8
- Al-Twairesh N, Al-Khalifa H, Al-Salman A, Al-Ohali Y (2017) Arasenti-tweet: A corpus for arabic sentiment analysis of saudi tweets. Procedia Comput Sci 117:63–72
- 24. Kora R, Mohammed A (2019) Corpus on arabic egyptian tweets. Harv Dataverse
- Al-Azani S, El-Alfy ES (2018) Emojis-based sentiment classification of arabic microblogs using deep recurrent neural networks. In: 2018 international conference on computing sciences and engineering (ICCSE). IEEE, pp 1–6
- Mourad A, Darwish K (2013) Subjectivity and sentiment analysis of modern standard arabic and arabic microblogs. In: Proceedings of the 4th workshop on computational approaches to subjectivity, sentiment and social media analysis. pp 55–64
- 27. Rosenthal S, Farra N, Nakov P (2019) Semeval-2017 task 4: Sentiment analysis in twitter. arXiv preprint arXiv:191200741
- Khalil EAH, Houby EME, Mohamed HK (2021) Deep learning for emotion analysis in arabic tweets. J Big Data 8:1–15
- Abdullah M, Shaikh S (2018) Teamuncc at semeval-2018 task 1: Emotion detection in english and arabic tweets using deep learning. In: Proceedings of the 12th international workshop on semantic evaluation. pp 350–357
- Mohammad SM, Salameh M, Kiritchenko S (2016) How translation alters sentiment. J Artif Intell Res 55:95–130
- Alharbi A, Kalkatawi M, Taileb M (2021) Arabic sentiment analysis using deep learning and ensemble methods. Arab J Sci Eng 46(9):8913–8923
- 32. ElSahar H, El-Beltagy SR (2015) Building large arabic multi-domain resources for sentiment analysis. In: International conference on intelligent text processing and computational linguistics. Springer, pp 23–34
- Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. arXiv preprint arXiv:13013781
- Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: International conference on machine learning. PMLR, pp 1188–1196
- 35. Pennington J, Socher R, Manning CD (2014) Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp 1532–1543
- Altowayan AA, Tao L (2016) Word embeddings for arabic sentiment analysis. In: 2016 IEEE International Conference on Big Data (Big Data). IEEE, pp 3820–3825
- Bojanowski P, Grave E, Joulin A, Mikolov T (2017) Enriching word vectors with subword information. Trans Assoc Comput Linguist 5:135–146
- Soliman AB, Eissa K, El-Beltagy SR (2017) Aravec: A set of arabic word embedding models for use in arabic nlp. Procedia Comput Sci 117:256–265
- Fouad MM, Mahany A, Aljohani N, Abbasi RA, Hassan SU (2020) Arwordvec: efficient word embedding models for arabic tweets. Soft Comput 24(11):8061–8068
- LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD (1989) Backpropagation applied to handwritten zip code recognition. Neural Comput 1(4):541–551
- Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
- Rehman AU, Malik AK, Raza B, Ali W (2019) A hybrid cnn-lstm model for improving accuracy of movie reviews sentiment analysis. Multimedia Tools Appl 78(18):26597–26613
- 43. Yang Q, Alamro H, Albaradei S, Salhi A, Lv X, Ma C, Alshehri M, Jaber I, Tifratene F, Wang W, et al (2020) Senwave: monitoring the global sentiments under the covid-19 pandemic. arXiv preprint arXiv:200610842
- 44. Ameur MSH, Aliane H (2021) Aracovid19-ssd: Arabic covid-19 sentiment and sarcasm detection dataset. arXiv preprint arXiv:211001948
- Mubarak H, Hassan S, Chowdhury SA, Alam F (2022) Arcovidvac: Analyzing arabic tweets about covid-19 vaccination. arXiv preprint arXiv:220106496
- Khanna VV, Chadaga K, Sampathila N, Prabhu S, Chadaga R, Umakanth S (2022) Diagnosing covid-19 using artificial intelligence: A comprehensive review. Netw Model Anal Health Inform Bioinforma 11(1):25

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.