SMISHING DETECTION APPLICATION BY USING AI

Hanan Alossimi, Nouraalotaibi, Alhnouf alsubaie, Hanan Aljuaid

Department of Computer and Information Science, Princess Nourah Bint Abdul Rahman University, Riyadh, Saudi Arabia

ABSTRACT

Smishing, or SMS phishing, is a major global mobile security concern due to the lack of spam filtering in SMS. To address this, our project aimed to develop a smishing detection application for Arabic SMS messages. We created "Etiqa," an application utilizing a hybrid CNN-LSTM deep learning model, which achieved 98% accuracy in classifying Arabic SMS messages. The dataset was meticulously collected and processed with natural language analysis tools tailored for Arabic. The algorithm was implemented in Python, and the user interface, designed for Android, was developed using Dart on the Flutter framework. The interface was integrated with the model via Fast API. Looking ahead, we plan to enhance the system's effectiveness and expand its capabilities.

KEYWORDS

Smishing, fraud, SMS, Detection, Artificial Intelligence

1. INTRODUCTION

The Arabic language, known for its distinct linguistic features, poses unique challenges for effective smishing detection. Current detection methods often fall short in recognizing the subtleties of Arabic text, resulting in less-than-ideal performance when it comes to spotting harmful messages.

To tackle this challenge, this study introduces a smishing detection application called "Etiqaa," which employs a hybrid model combining CNN and LSTM specifically designed for Arabic text. Convolutional Neural Networks (CNNs) are skilled at identifying local features, such as n-grams and character sequences, which are vital for recognizing patterns in Arabic. Meanwhile, Long Short-Term Memory (LSTM) networks are proficient in modeling long-term dependencies, which are crucial for grasping the context and intent behind a message.

Key contributions of this paper include:

- **Enhancing accuracy:** Utilize the strengths of CNNs and LSTMs to improve smishing detection effectiveness in Arabic.
- **Capturing linguistic subtleties**: **Successfully** address the unique aspects of Arabic text, including its morphological and syntactic features.
- **Managing varying message lengths**: Adapt to **the** different lengths of smishing messages, ensuring reliable performance across various attack types.

David C. Wyld et al. (Eds): NLAI, IBCOM, MLDS, GridCom, SCAI, ICCSEA, SPPR – 2024
pp. 41-51, 2024. - CS & IT - CSCP 2024 DOI: 10.5121/csit.2024.142204

- **Developing an APPLICATION:** The application can retrieve **incoming** messages, classify them, and provide alerts for spam messages, along with a user-friendly guide on reporting phishing attacks to the authorities.

This research aims to advance smishing detection methods, particularly for the Arabic language, by offering a strong and efficient model that can help reduce the risks associated with this increasing threat, all within a user-friendly application.

2. RELATED WORK

In a comparative study of text classification algorithms for phishing message detection, the proposed research [1] has tested several classification algorithms for the classification of phishing messages in Arabic and English. The results revealed that the CNN-LSTM hybrid model had the best accuracy among several models that have been tested, such as SVM, K-Nearest Neighbors, Multinomial Naive Bayes, Decision Tree, Logistic Regression, Random Forest, AdaBoost, Bagging classifier, Extra Trees, CNN, and LSTM.Taking that into consideration, we built our model for the Smishing detection application as a CNN-LSTM hybrid model.

DSmishSMS-A [2]: In this paper provides a methodology for detecting SMS phishing that consists of two phases: domain checking phase and SMS classification phase. As a key component of this model, they looked at the legitimacy of the URL in the SMS. The system then uses the Backpropagation Algorithm to classify the messages. The evaluation's results showed a 97.93% accuracy rate, demonstrating how effective the suggested strategy is for smishing message identification. However, creating a signature of minimal information that an attacker shares in a message is a difficult task and the phone number and email ID included in a message are not checked for their maliciousness.

SMS filter[3]**:** this app provides features for users such as creating their own list of rules to filter incoming text messages , it filters incoming text messages by text patterns, keywords, regexp.
blacklist or whitelist messages that match particular rule, also user can manage the blacklist and whitelist ,by adding some keywords to the two lists. Whitelist allows the important message,and the blacklist ignores the message which is spam.

Table 1: Similar System comparison.

Features	our system	Smishing Detector [4]	Cantina+[5]	Rule-based[6]	"SmiDCA"[7]
Operating platform	Android	Android	Microsoft IE extension	WEKA software	Not mentioned
Used techniques and Algorithms	Deep learning algorithms such as: CNN, LSTM Hybrid model	Naive Bayes Classification Algorithm is used to identify the malicious contents and keywords present in the text message.	4 sets of heuristics were used and evaluated under some circumstances to prove efficiency, TF-IDF	a rule-based approach for the classification and detection of smishing, Decision Tree, RIPPER, and PRISM, Ripper outperformed with high accuracy	classifiers: Random Forest, Decision Tree, Support Vector Machine, and AdaBoost, correlation algorithms
Accuracy	98%	96.29%	97%	95.92%	96.4% for the English dataset 90.33% for the non-English dataset
Supported language	Arabic	English	English	English	English, non-English
Free	Free to use	Free to use	Free	Free source	Not free
Dataset	Dataset is collected from Kaggle and hugging face	dataset is collected from a proposed work by the author Almeida	used an already compiled list of word frequencies based on the British National Corpus	dataset v.1 is collected from SMS Spam research work [8]	English and non-English datasets
Programming language	Python - Dart	Python	C#	Python	Python

3. METHODOLOGY

Taking into consideration the many problems that occur through phishing, we suggest developing an application that solves the problems of fraudulent messages. It is based on classifying messages based on whether they are fraudulent or not by using the artificial intelligence algorithms that CNN and LSTM hybrid models in this section will describe in detail.

3.1. Model

The core idea of this detection system is to analyse collected SMS messages and apply machine learning techniques to classify them as phishing or non-phishing. The process begins by cleaning up unnecessary data found in the text. Then, a pre-processing step is performed to represent the textual data in a format suitable for input to the machine learning models.

3.1.1. The Data Used

The dataset that has been used in this project is an Arabic SMS-spam data set that was collected from various resources, such as Kaggle and Hugging Face. The dataset size is 5723.

The unequal distribution of classes in this initiative is one of its main obstacles. Class imbalance arises when one class, the minority group, has substantially lower samples than the other class, the majority group, in a binary classification issue involving data samples from two groups. In numerous issues Due to the majority group's higher prior probability, learners tend to overclassify when there is a class imbalance in the training set. This leads to a higher rate of misclassification of occurrences belonging to the minority group than to the majority group.[9]

SMOTE (Synthetic Minority Over-sampling Technique) is an oversampling technique that generates synthetic samples for the minority class to solve this issue with data augmentation. This algorithm aids in resolving the random oversampling-induced overfitting issue. It employs interpolation between positively aligned instances to create new instances by concentrating on the feature space. This facilitates the machine learning model's ability to more successfully identify patterns and traits unique to the minority class. [10]

3.1.2. Data Cleaning

There are several different techniques that can be used for data cleaning, and the best technique will vary depending on the specific dataset. However, some common techniques include:

Removing unimportant data: it’s important to consider removing unimportant data since it can often be noise in a dataset as it does not provide any information about the meaning of the text, it can affect the model performance, for example: punctuation, digits, English words, new lines,emojis

Removing stop words: stop words, in our case, are unimportant features during classification. Normalization: involves converting all words to a common formincluding:Unicode characters in a string the Alef Maksura characterreplaces different forms of the Hamza character for example (آ- إ- أ) to (ا)Arabic text with TehMarbuta character in example (ة) to (ه) .This can help to improve the accuracy of the model, as it will not be confused by different forms of the same word.

3.1.3. Text Preparation

One of the common practices in natural language processing (NLP) used to prepare text data for machine learning models is word embedding, where words are represented as vectors in a continuous space, capturing many syntactic and semantic relations among them. However, deep learning models are not able to understand natural languages as they are; they are only capable of representing numerical data. the suggested method makes use of an open-source Arabic word embedding model called AraVec[11],which is a Twitter Skip gram pre-trained distributed word embedding model with total embedding dimension=300 that attempts to offer robust and free word embedding models to the Arabic NLP research community. The models we have developed

were built with the help of the Gensim tool, an effective toolkit designed to handle numerous common NLP tasks, one of which is an implementation for the Word2Vec model.

In thisproposed approachthe first stage in this suggested method converts the text in the datasets to sequences of integers by the Tokenizer tool. Then ensure that all the sequences have the same length by adding padding tokens at the end. The max_length variable is set to 30, which means that all documents will be padded to a length of 30 words by the pad_sequences function is also a powerful tool, By doing this, the text data is guaranteed to be in a format that the input for word embedding.By using the Gensim library loads model Twitter Skip Gram and initiate the dictionary The KeyedVectors.load() function loads the word2vec. So, for each word in the loaded model's vocabulary (w2v_model.wv), its corresponding vector representation is added as a value to the w2v_embeddings_index dictionary, with the word itself as the key by using a for loop and creating matrix for maps each word in the training data to a unique integer index) and w2v_embeddings_index dictionary is retrieved and added to the embedding_matrix if it exists if not keep the value zero.

3.1.4. Deep learning-based model

Although the Convolution Neural Networks algorithm is commonly used for image classification, CNNs can also perform well in text classification based on the one-dimensional convolutional idea. However, since CNNs are not capable of correlating current information with past information, they are often combined with LSTMs to address this issue. This is where each layer of the model is explained for clarity. The model type is the Sequential model. That means each layer serves the next layer in order

- The input layer: The model begins with the placeholder layer, where the maximum length value is the same and the input shape is = 30 for length sequences.several semantic and grammatical relationships between them.
- Embedding layer: The embedding layer uses the input layer as input and converts the integer inputs into a dense vector representation.
- Convolutional 1D layer: Convolutional One-Dimensional Layer This layer is to extract relevant features from text data. And it is the most important layer for understanding how it works. Let us know what The input argument of this layer. One is that it has 32 filters that help to identify diverse patterns or features in the vector representation. Also, the size of Kernal 3 specifies that each filter slides over a sequence of three consecutive words at a time; in other words, it corresponds to the size of the convolution window. in addition The value of padding is the same argument to ensure that the output sequence length is the same as the input sequence length. And activation equal ('relu') specifies that the ReLU activation function should be used. ReLU introduces non-linearity and helps the model learn more complex patterns.
  Extract the features, done by Applying the convolution operation to words resulted in a dense vector representation. First, the layer uses a window of three-word embeddings in a row. Next, it modifies this window of word embeddings using a set of thirty-two learnable filters. In our case, dimensionality = 300, each filter is a vector of weights with the same dimensions as the word embeddings. The result is saved in the matching element of the output feature map. The dot product between the filter and the window of word embeddings is computed. Once the full input sequence has been analyzed, one-word embedding is used to shift the window, and the procedure is repeated.
- max pooling layer: The most important function of the convolution one-dimensional layer is to extract relevant features from text data. The input argument of this layer is the number of filters, which is 32, that learn to detect different patterns or features in the

word embeddings. The kernel size of 3 specifies that each filter slides over a sequence of three consecutive words at a time.

- LSTM : One kind of recurrent neural network (RNN) that can learn long-term dependencies in sequences is the LSTM (Long Short-Term Memory) layer. The LSTM layer plays a key role in increasing the model's accuracy. The LSTM layer has 64 units and an activation function of ReLU due to CNN's incapacity to connect previously acquired information to freshly acquired information. Dropout=0.2, which randomly sets 20% of the input units to 0 during training, aids in regularizing the model. Because the words in the input sequences must appear in the correct order, the objective is to capture the temporal connections between the words in the sequences. The data's intricate relationships and patterns can be identified by the LSTM layer.
- Dense layer: The final output of classification which 0 ,1 come with Dense layer and dense =1 means the is binary classification, with a sigmoid activation function is used for binary classification.

3.1.5. Model training and testing

There are several techniques for model training, for example. Hold-out, K-folds, Leave-one-out In our model, we used a 5-fold cross-validation approach for data training, which means that the dataset was divided into five specific groups, or, in other words, fold. Every single fold is designated as a testing set while using the remaining K-1 as the training set. This technique helps to mitigate the potential bias introduced by a single train-test split and provides a more reliable estimate of the model's performance. This was achieved using the KFold class from the sklearn.model_selection module, which is for the creation of the important training and testing indices for each fold.

3.2. Etiqaa App

The app was created to be user-friendly. We used the Dart programming language to build interfaces in Flutter Framework. Flutter framework provides a telephone package to read messages from the Messages application. We used Android Studio, flutter and Visual Studio Code

The app was designed to work as follows:

- We get permission from the user using the "get_permission function", with an internal variable called "state" that stores "allow.sms.request" to ask the user to access the messages, then we check the permission with an if statement. If the result is true, we read the messages using the "fetch message function" SMS listener which is in flutter “Telephony package”.

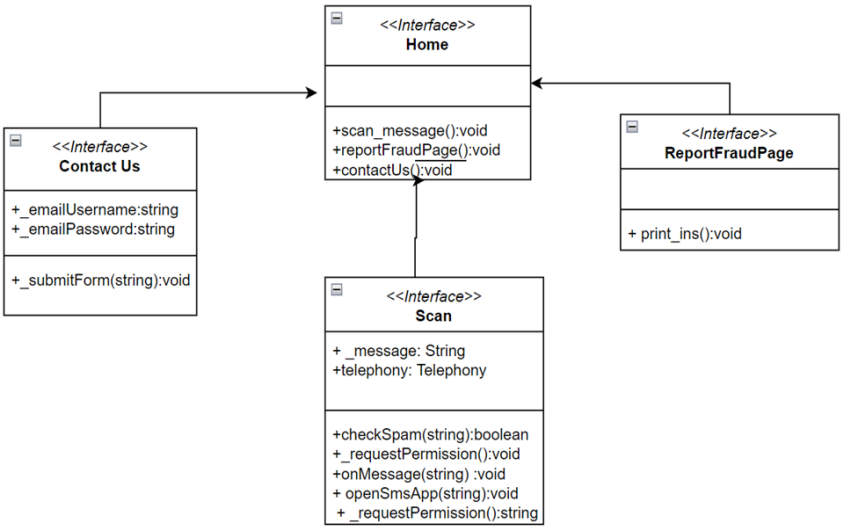
- Then we made a list called "SmsMessage", which takes the message from the telephone package, and sends each message with its index to a function called spam "check spam" which has an API.

- We prepare a request and send the message as a string to the API, then it returns a true or false result. Based on the result, a specific format will appear in the examination interface, if it is fraudulent, it will appear in red with a warning sign next to it.

- When Clicking on the message it will open the SMS app by "openSmsApp" function. We also made the application run in the background by function "listenForIncomingMessages

In addition, the app have a guide for users on how to report spam messages to the authorities.

Also,the user can communicate with the support team and provide feedback.



3.3. Deploy the Model as API

After training and testing the model, we Deployed the CNN-LSTM model as an API by using the FastAPI framework to make the model accessible to our application.
We followed the following steps:

Saving and loading the pre-trained model: Use the save() method of a TensorFlow Keras model to save the pre-trained model in HDF5 format. Later, load the saved model using the tf.keras.models.load_model() method to make predictions. Saving the tokenizer using the pickle library ensures that the same tokenizer is used during prediction as during training. Create a function that performs all steps including to data cleaning and all preprocess messages and , such as normalizing Arabic text, removing stop words ..esc Define an endpoint using the @app.post decorator from FastAPI that accepts HTTP POST requests to the /Classification endpoint. POST request, to makes predictions on the pre-processed input data. Return the predictions in a list of dictionaries, where each dictionary contains a label ('spam' or 'ham') and a confidence score. Finaly, we used Postman tool for testing the API.

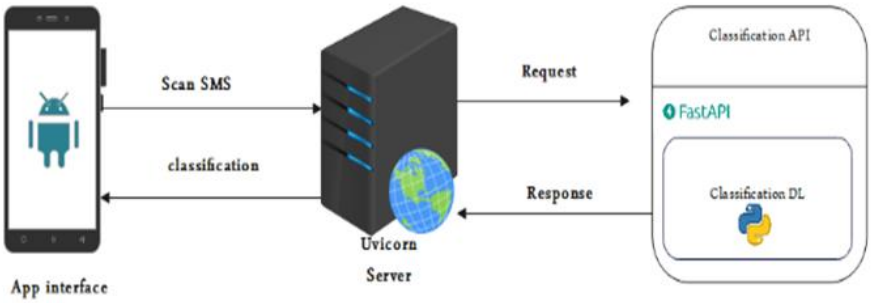


Figure 2 Application architecture

4. EVALUATION

The results of the model's performance help us make decisions about whether the model is good or bad and to prove the robustness of this model in classification. Here is an estimate of the model's overall effectiveness. The mean accuracy of 98.53% is exceptionally high, which means the model performed very well overall. The mean loss of 5.04% is also quite low, highlighting the robust performance of the model even more.

4.1. Confusion Matrix

Our dataset contains 5078 messages 581 are spam and 4497 are Non-spam, The first step to evaluate the performance of the CNN-LSTM model will apply the Confusion Matrix technique to allow us to measure recall, precision, accuracy, and AUC-ROC curve, it will clarify how our model performed on a binary classification spam and non-spam, Additionally, it is a reliable approach for determining how well the model classified each data and any potential inaccuracies. The matrix columns reflect the results, while the matrix rows represent the actual labels in the training dataset.

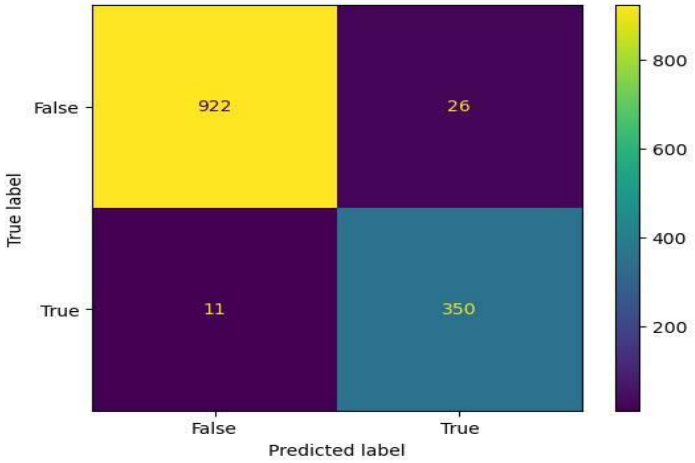


Figure 3 confusion matrix

4.2. ROC curve and AUC

The ROC is evaluating performance across all possible classification thresholds, It reveals how well the model can differentiate across classes, The main purpose to Analysing the strength power of a classifier

In our classification model is to assign higher probabilities to observations that belong to class spam and lower to class Non spam, Basically, we will be able to select a threshold and forecast the two classes with high confidence if there is a significant difference in the probabilities given to observations of either class.

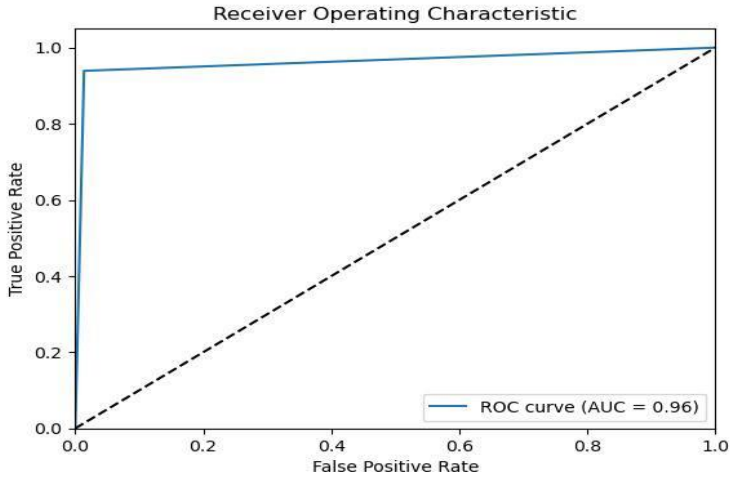


Figure 4 ROC

The ROC as show in plot is closer to the top left of the graph, so we can evaluate it as good model. as well as AUC value of 0.96 is near to the 1 which means it has a good measure of separability and in general can correctly identify a large proportion of positive cases because it has high true positive rate (TPR) while minimizing the number of false positives (FPR).

4.3. Model Testing with Spam Messages

```
text= ['05799430 التالي  الرقم  مع بالتواصل سارع الراجحي بنك في حسابك اختراق تم لقد']
# preprocess the text data
encoded_text = t.texts_to_sequences(text)
print("encoded_text:\n",encoded_text)
# pad documents to a max length of 30 words
padt = pad_sequences(encoded_text, maxlen=max_length, padding="post")
print("padded_text:\n",padt)
# make predictions using the loaded model
predictions = CNN_LSTM_model.predict(padt )
# decode the predictions into class labels
#predicted_labels = decode_predictions(predictions)
print("predictions",predictions )
```

```
encoded_text:
 [[2, 45, 4856, 542, 3926, 9328, 652, 151, 303]]
padded_text:
 [[   2   45 4856  542 3926 9328  652  151  303    0    0    0    0    0
     0    0    0    0    0    0    0    0    0    0    0    0    0    0
     0    0]]
1/1 [==============================] - 0s 17ms/step
predictions [[1.]]
```

Figure 5: Code for pre-trained model.

5. RESULT

In our project, Different types of testing were done on our application.Unit testing was conducted while working on the program and the algorithm as separate units to make sure that it operates efficiently , then we moved to the stage of integration testing, in addition, the functional testing and end-user testing were conducted , as well as we test All possible cases Whether in algorithm performance or application interfaces, Which proved the effectiveness of the application and its ease of use, and the goal of the application was achieved with the efficiency of the hybrid CNN-LSTM classification algorithm. The experimental results showed that our model achieved an

accuracy of 98% and AUC of 95%, and our application Etiqa was able to meet the requirements that were previously specified, in addition to that the system interfaces are simple and easy to use.

6. CONCLUSION

In conclusion, after a lot of research and comparison with our goals in the project, we aimed to address the growing concern of smishing attacks on mobile devices by developing a detection application using artificial intelligence algorithms. The project recognized that with the increasing use of technology in our lives, there is a high risk of personal information being exposed to attackers, and smishing is one of the most significant mobile security concerns globally.

The project's proposed solution of developing a smishing detection application has the potential to provide users with a reliable and effective tool to protect themselves from smishing attacks. The use of artificial intelligence algorithms can help improve the accuracy and efficiency of the detection system, making it easier for users to identify and avoid smishing messages.

However, the success of the project depends on the accuracy and reliability of the detection system, which can be challenging to achieve given the constantly evolving nature of smishing attacks. Therefore, ongoing research and development are needed to ensure the system remains effective in detecting new and emerging smishing threats.

ACKNOWLEDGEMENTS

At the outset, we would like to thank God who has given us the grace to reach this level. we are grateful for the efforts you have made for us (College of Computer and Information Sciences), and in particular, (Department of Computer Science) .We owe great gratitude to our university, Princess Noura bint Abdul Rahman. A university that we are proud of, that helped develop us as computer scientists and provided us with the tools we need to make our transition to this stage as easy as possible. Our supervisor, Dr. Hanan Al-Juaid, deserves special thanks for her time, effort and patience with us. Which was the first supporter and source of inspiration. We are grateful that she is our supervisor.We express our sincere appreciation to our family, who have always been there for us, as well as our teachers and mentors.

REFERENCES

[1] A. Ghourabi, M. A. Mahmood and Q. M. Alzubi, "A Hybrid CNN-LSTM Model for SMS Spam," Future Internet, 2020.
[2] M. Sandhya and S. Devpriya, "DSmishSMS- A System to Detect Smishing SMS," Neural Comput & Applic, 2021.
[3] I. Mozharovskyi, "SMS filter," App store, 2021.
[4] M. Sandhya and S. Devpriya , "Smishing Detector: A security model to detect smishing through SMS content analysis and URL behavior analysis," 2022.
[5] X. GUANG, H. JASON, P. R. CAROLYN and C. LORRIE, "Cantina+: A Feature-Rich Machine Learning Framework for Detecting Phishing Web Sites," 2011.
[6] K. J. Ankit Kumar and B. G. B., "Rule-Based Framework for Detection of Smishing Messages in Mobile Environment," Procedia Computer Science, 2018.
[7] K. S. Kuppusamy and G. Sonowal, "SmiDCA: An Anti-Smishing Model with Machine Learning Approach," The Computer Journal, 25 4 2018.
[8] T. Almeida, . H. J. Gomez and A. Yamakami, Contributions to the Study of SMS Spam Filtering: New Collection and Results, 2013.

[9] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," Journal of Big Data, 19 March 2019.
[10] S. SATPATHY, "SMOTE for imbalanced Classification with Python," Analytics Vidhya, 5 8 2024.
[11] S. Abu Bakr, E. Kareem and S. R. El-Beltagy, "“AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP”," in 3rd International Conference on Arabic Computational Linguistics (ACLing), Dubai, 2017.