Hindawi
Journal of Electrical and Computer Engineering
Volume 2024, Article ID 4976944, 11 pages
https://doi.org/10.1155/2024/4976944

Research Article

Convolutional Neural Networks to Facilitate the Continuous Recognition of Arabic Speech with Independent Speakers

Sally A. Sayed,[1] **Rania Ahmed Abdel Azeem Abul Seoud**,[2] **and Howida Y. Abd El Nady**[1]

[1]*Department of Computer Science, Faculty of Computers & Artificial Intelligence, Fayoum University, El Fayoum 63514, Egypt*
[2]*Department of Electrical Engineering, Faculty of Engineering, Fayoum University, El Fayoum 63514, Egypt*

Correspondence should be addressed to Sally A. Sayed; sa1828@fayoum.edu.eg

Received 21 December 2023; Revised 25 March 2024; Accepted 13 April 2024; Published 29 April 2024

Academic Editor: Mustafa Sameer

Copyright © 2024 Sally A. Sayed et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Automatic speech recognition (ASR) is a field of research that focuses on the ability of computers to process and interpret speech feedback from humans and to provide the highest degree of accuracy in recognition. Speech is one of the simplest ways to convey a message in a basic context, and ASR refers to the ability of machines to process and accept speech data from humans with the greatest degree of accuracy. As the human-to-machine interface continues to evolve, speech recognition is expected to become increasingly important. However, the Arabic language has distinct features that set it apart from other languages, such as the dialect and the pronunciation of words. Until now, insufficient attention has been devoted to continuous Arabic speech recognition research for independent speakers with a limited database. This research proposed two techniques for the recognition of Arabic speech. The first uses a combination of convolutional neural network (CNN) and long short-term memory (LSTM) encoders, and an attention-based decoder, and the second is based on the Sphinx-4 recognizer, which includes pocket sphinx, base sphinx, and sphinx train, with various types and number of features to be extracted (filter bank and mel frequency cepstral coefficients (MFCC)) based on the CMU Sphinx tool, which generates a language model for different sentences spoken by different speakers. These approaches were tested on a dataset containing 7 hours of spoken Arabic from 11 Arab countries, covering the Levant, Gulf, and African regions, which make up the Arab world, and achieved promising results. CNN-LSTM achieved a word error rate (WER) of 3.63% using 120 features for filter bank and 4.04% WER using 39 features for MFCC, respectively, while the Sphinx-4 recognizer technique achieved 8.17% WER and an accuracy of 91.83% using 25 features for MFCC and 8 Gaussian mixtures, respectively, when tested on the same benchmark dataset.

1. Introduction

Arabic is recognized as the official language in 22 countries globally, with approximately 400 million individuals estimated to be proficient in speaking Arabic across the world [1]. Arabic is a collection of languages that are closely related. Modern standard Arabic (MSA) is frequently taught in schools and colleges and is used in formal settings such as courtrooms, media, and official speeches. All countries that speak Arabic speak MSA as their official language [2]. Despite being one of the most widely spoken languages globally and ranking fifth in terms of usage [3, 4], research studies on speech recognition for the Arabic language are limited [3, 4]. Arabic ASR poses several challenges, including sparse language data, lexical variety [5], different dialects spoken throughout the Arab world [6], and the widespread use of nondiscretized text [4]. Furthermore, the Arabic language presents difficulties for the speech research community due to its morphological complexity [7] and the complexity of large vocabulary ASR.

Diacritical and nondiacritical texts are the two types of writing that are available in Arabic. Diacritical marks are symbols that are essential for understanding a word's meaning. Each Arabic word's specific definition is determined by its diacritical mark. Arabic readers and speakers can read and understand nondiacritical material by using the word's context in a phrase [8, 9].

Automatic speech recognition (ASR) has emerged as a powerful algorithm for simplifying human-computer interaction (HCI) by serving as the most interactive and convenient means of communication between automated systems and individuals [10].

The conventional structure of automatic speech recognition (ASR) algorithms involves various components, including lexicon creation, language modelling, and acoustic modelling. These components are constructed and processed independently of each other [11]. The traditional approach for ASR includes a variety of methods, such as deep neural networks (DNN), hidden Markov models (HMMs), Gaussian mixture models (GMMs), and a hybrid of HMM and DNN [12]. On the other hand, with the end-to-end strategy, all components can be trained and adjusted together as a single entity through the use of deep learning techniques. This approach is generally viewed as more advantageous than the conventional approach in terms of the optimization of all its components [11].

The end-to-end approach does not utilize any linguistic knowledge in its data, and all of its components may be trained using a single algorithm [12]. Various neural models and properties have been studied by researchers to explore the workflow processes involved in this approach [12, 13]. The entirety of automatic speech recognition, or ASR, is composed of a variety of techniques and methods, employing a range of techniques, such as recurrent neural networks (RNNs) transducer and connectionist time classification (CTC), convolutional neural networks (CNNs), long short-term memory networks (LSTM), attention-based encoder-decoder, and hybrid models [14].

Due to the availability of spectrum fluctuations and local correlation characteristics in speech signals, the CNN technique is suitable for ASR. Local filtering and weight-sharing techniques are used in CNN's pooling layer to efficiently remove feature noise and shift tiny frequencies.

In addition, LSTM is used as a cutting-edge approach for modelling acoustic data, which can result in higher recognition rates. The LSTM has a memory block that enables it to control the learning time dependencies brought on by disappearing and exploding gradient occurrences. As a result, end-to-end algorithms now regard the hybrid CNN-LSTM approach as a cutting-edge method.

The proposed paper presents two techniques. The first involves the hybridization of CNN-LTSTM and the attention-based encoder-decoder to develop a new method for decoding nondiacritics Arabic ASR. The second technique is traditional and involves utilizing the Sphinx-4 recognizer, which comprises a pocket sphinx, sphinx base, and sphinx train, with various types and amounts of feature extraction (filter bank and MFCC). Furthermore, we used the CMU Sphinx tool to develop a language model that includes uni-grams, bi-grams, and tri-grams for different speakers with different sentences to attain each technique's intended result. Based on the experiments conducted, our work demonstrates better accuracy and performance for Arabic ASR algorithms compared to traditional ASR algorithms.

The remaining parts of the paper are organized as follows: Section 2 gives a brief overview of the Arabic ASR research that uses several techniques, including end-to-end Arabic ASR that has been around for a while, HMM-based neural networks, recurrent neural networks, and deep learning. It will then go on to research methodology in Section 3, while the fourth section presents the findings of the research. Lastly, Section 5 discusses the conclusion and prospects.

2. Related Work

Statistical models used in automatic speech recognition have significantly improved the recognition of speech in various languages [15]. A statistical speech recognizer is composed of the following elements:

(1) A language model that calculates the likelihood of a sequence of words.

(2) An acoustic model that specifies the correlation between acoustic data and phonemes.

(3) A lexicon is necessary for the decoder to establish the phonetic sequence for each word [16].

(4) The decoder identifies the sentence with the highest likelihood, based on the input acoustic observations. This can be achieved by multiplying the two probabilities for each sentence and selecting the sentence with the highest product [17].

Khatatneh [18] introduced an algorithm for recognizing Arabic phonemes and employed various techniques to enhance the accuracy of the algorithm. The preprocessing phase included the integration of the Gaussian low-pass filtering algorithm with a neural network. In addition, the neural network was utilized to train the speech signal recognition algorithm. Sampling, signal capture, energy setting, and quantization were among the techniques employed to recognize phonemes. They concluded that utilizing a neural network led to improved results.

A method for automatic Arabic speech recognition's Egyptian dialect was proposed by A. Mousa et al. [19]. They successfully represented the features by combining feature-rich modelling (DNN-LMs) with morpheme LMs. In addition; they combined words and morphological elements with the corresponding features. The results of the study demonstrated that this method was more accurate than traditional word-based (LMs).

In the same direction, AlHanai et al. [20] developed an automatic Arabic speech recognition algorithm utilizing a speech corpus of 1200 hours. They utilized a deep neural network (DNN) that integrated various techniques, including feed-forward, time delay, convolutional, Highway LSTM (HLSTM), Recurrent (LSTM), and Grid LSTM (GLSTM). The research revealed that the GLSTM model, trained on the chosen corpus, achieved a WER of 18.3%.

An Arabic news voice recognition algorithm was created by Ali et al. [21]. They created the algorithm using the KALDI recipe and trained the acoustic model for 200 hours using the broadcast news algorithm. Text normalization, vowelization, and language models were all included in their evaluation of the system using BR, BC, and combination metrics. The results showed that, for BR, BC, and combined

metrics, the algorithm attained WER scores of 15.81%, 32.21%, and 26.95%, respectively.

AbdAlmisreb et al. [22] proposed a deep neural network (DNN) approach to construct an automatic Arabic speech recognition system. The DNN featured three hidden layers, 500 maxout units, and two neurons per unit with features generated by MFCC. The study employed a corpus of 20 Malay speakers of Arabic phonemes, with five waveforms used for training and fifteen for testing. The Maxout-based deep structure outperforms several other deep networks, including CNN, restricted Boltzmann machines, conventional feed-forward neural networks, deep belief networks, and convolutional auto-encoders, according to the results.

Ahmed et al. [17] applied recurrent neural networks (RNNs) without a lexicon, employing the objective function of connectionist temporal classification (CTC), which utilizes audio input information. They used a dataset containing 1200 hours of Al Jazeera's multigenre broadcast program to assess the recipe, resulting in a WER of 12.03%. It is worth noting that this study employed various dialects, whereas Ahmed's study used only one dialect.

Azim et al. [23] created a phonetic decision tree for Arabic. They utilized their method to create tied-state triphone hidden Markov models (HMMs) and compared the experimental results with data-driven tri-phone models and phoneme-based models. Their approach achieved a maximum accuracy of 95.13%. When tested on the same benchmark database, the accuracy was reduced to 78.03% and 58.45% using tri-phones and HMMs based on phonemes, respectively, employing data-driven techniques. The dataset utilized in this study consisted of 4372 WAV files, totalling 7 hours of recordings from a single Arabic speaker. It is noteworthy that although Azim's study included multiple speakers and sentences, they only used one speaker.

Alsayadi et al. [24] introduced cutting-edge methods to enhance discrete automatic Arabic speech recognition, including CNN-LSTM, CTC-based ASR, and an attention-based end-to-end approach. To improve the outcomes, a language model based on words is employed. To train and test the framework, SASSC—the standard Arabic single-speaker corpus—was employed. According to the results of the experiment, the CNN-LSTM attention framework was more effective at recognizing Arabic speech when compared to conventional ASR, as well as when compared to the common CTC-attention ASR framework. The CNN-LSTM with an attention framework resulted in a significantly lower word error rate (WAR) compared to the CNN-LSTM framework (5.24% vs. 2.62%).

3. Methodology

3.1. The Proposed Algorithm. This study utilized two speech recognition techniques—the CNN-LSTM attention technique and the Sphinx-4 technique. All algorithms applied an acoustic model that was trained on the speech corpus, and additional information about these algorithms will be provided in subsequent sections. The language models were developed using the CMU Sphinx model, and the hardware specifications utilized are listed in Table 1, and the main steps of this research are shown in Figure 1. The initial step of the data preprocessing involves extracting features from the sounds and constructing a language model, which is then fed into the recognizer for evaluation using the character error rate, word error rate, and accuracy equations. This comprehensive approach ensures thorough analysis and assessment of the extracted features and their performance within the recognition system.

Table 1: Hardware specifications.

Table 1	Machine hardware specifications
CPU	Intel® Core™ i7-6820HQ CPU @ 2.70 GHz × 8
GPU	Intel® HD graphics 530 (SKL GT2)
RAM	8 GB
CUDA version	9.1
OS	Ubuntu 18.04

3.2. Data Preprocessing. The initial stage involves preparing the dataset by converting the file format from Wav to Flac, which helps to reduce the storage space while maintaining high quality. In addition, the vocabulary file is created by extracting all words from the sentences and partitioning them into subwords with the aid of the SentencePiece Library. The SentencePiece Library is composed of four primary parts: the normalizer, trainer, encoder, and decoder. The normalizer functions to standardize semantically similar Unicode characters into a standard form. On the other hand, by indicating the kind of subword model to be employed, the trainer trains the subword segmentation model from the normalised corpus. The encoder utilizes the normalizer and tokenizes the input text into a subword sequence utilizing the subword model learned by the trainer. The decoder eventually transforms the subword sequence into a normalised text [25].

3.3. Feature Extraction. This section delves into the topic of feature extraction, employing two distinct techniques. The first technique discussed is MFCC 4.3.1, which involves extracting features. The second technique explored is the filter bank 4.3.2, which also encompasses feature extraction. By examining and comparing these two approaches, a comprehensive understanding of the feature extraction process is achieved.

3.3.1. MFCC Feature Extraction. As waveforms cannot be directly processed by tools and speech modelling, they are transformed into a set of acoustical feature vectors. This research employs the MFCC feature extraction method, which involves a set of MFCC coefficients and energy measures. The study will use three sets of MFCC vectors, comprising 13 MFCC vectors (including 12 cepstral features and an energy measure), 25 MFCC vectors (containing the 12 cepstral features' change rates and the energy measure's change rate), and 39 MFCC vectors (which comprise 12 cepstral features, an energy measure, as well as the velocities and rates of change of these 13 features).

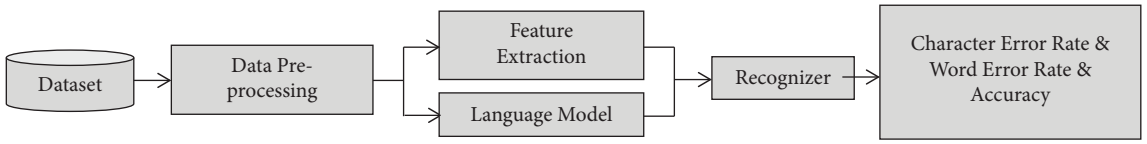

Figure 1: The main steps of this research.

3.3.2. Filter Bank Feature Extraction. The filter bank representation serves as a simplified model, crucially simulating the initial stages of transduction in the human auditory system, thereby playing a vital role in speech processing. This concept is supported by two significant factors. Firstly, according to the "place theory," the maximum displacement location along the basilar membrane, in response to pure tones, correlates directly with the logarithm of the tone's frequency. Secondly, extensive studies on human perception have demonstrated the challenge of distinguishing frequencies of complex sounds within a specific bandwidth surrounding a nominal frequency, unless one of the sound components exceeds that bandwidth. This range of frequencies is referred to as the critical bandwidth [26, 27].

In light of these findings, the proposed paper adopts three sets of filter bank vectors, encompassing 40 vectors, 80 vectors, and 120 vectors, respectively. This choice allows for a more comprehensive exploration and analysis of the filter bank representation's effectiveness in speech processing.

3.4. CMU Sphinx Language Model. A language model (LM) is a probabilistic statistical model capable of generating fictional word sequences. In the realm of speech recognition, one commonly used language model is the *N*-gram model, which operates as a two-word model. This model takes into account a set of preceding words, predicts the probability of the next word, and assigns scores to the likelihood of generating complete sentences [28]. The language model plays a crucial role in system configuration, providing the decoder with information about potential word sequences for recognition. Various models, such as phonetic language models, statistical language models, and keyword lists, exhibit different performance characteristics. During runtime, you have the option to choose a decoding mode that meets your needs and can switch between modes. With the keyword spotting mode offered by PocketSphinx, you can specify the list of keywords to look for and set a threshold for each keyword to find it even in continuous speech. In comparison, other modes try to identify words based on the grammar, even if you use words that are not in it. PocketSphinx is the only algorithm that provides support for keyword lists, while Sphinx4 does not have this capability. Language models for sequential words use three types of probabilities: 1-gram, which indicates the likelihood of a single word in the text; 2-gram, which shows the probability of two words occurring together in the text; and 3-gram, which represents the probability of three consecutive words appearing together in the text [29].

The study utilized the CMU Sphinx toolkit as its language model, which employed *n*-grams. Initially, the default unigram count was computed, and then, a task vocabulary with word frequencies was created. Bi-grams and tri-grams were generated from the training text, based on this vocabulary. Finally, the *n*-grams were converted into a binary format, language model, and standard ARPA format.

3.5. Model Recognizer. This section provides an in-depth discussion of the model recognizer, encompassing three distinct techniques. The first technique involves employing CNN-LSTM architecture with MFCC 4.5.2 as the feature extraction method. The second technique explores the utilization of CNN-LSTM with filter bank 4.5.3 for feature extraction. Lastly, the third technique focuses on the implementation of the Sphinx-4 tool recognizer 4.5.4. By exploring and comparing these three techniques, a comprehensive analysis of the model recognizer is achieved, highlighting the strengths and capabilities of each approach.

3.5.1. CNN-LSTM Technique. The CNN algorithm is highly recognized and extensively applied in the realm of deep learning. It possesses a unique advantage over earlier methods as it can identify crucial features without the need for human involvement. LSTM stands for long short-term memory, and it belongs to the category of recurrent neural networks (RNNs) but with a more intricate computational unit. LSTM is well-suited for processing inputs of varying lengths and excels at capturing intricate long-range dependencies due to the presence of a forget gate [30].

The CNN-LSTM model is powered by PyTorch, and the attention method is utilized in the sequential process of the model, as illustrated in the next sections. Preprocessing of corpus data and lexicon, as well as language modelling, is the initial step, followed by feature extraction. Language modelling is further trained and developed in the subsequent stages. The encoder is constructed using CNN and LSTM technology, while the decoder is constructed using attention technology. The data preprocessing and feature extraction phases were conducted by the procedures outlined previously. This model leverages the training and collected data to generate and train an external language model (LM). The external language model consists of approximately 262936 words with 28682 separate words. The CMU sphinx language model was used to extract the language model, as described in Section 3.4.

An encoder-decoder model has been developed for network training and recognition. The acoustic model is built using a hybrid CNN-LSTM method in the encoder section. The input sequences are transformed into vectors, using CNN layers and then fed into the LSTM [31]. A 4-layer architecture is employed, which applies 2D convolution and (3; 3) kernel size to the feature and time frame axis, which was utilized in previous studies [32, 33]. To achieve results in

frames that are one-fourth the time length after convolution, downsampling at layers 1 and 3 was applied, reducing the signal's sampling rate by a factor of $2x$. After passing through the stacked convolution layers, the output is then fed into three layers of bidirectional LSTMs [34]. Word encoding is performed one word at a time, relying on the stacked CNN and current status.

Following this, the status is updated and made available for subsequent steps. Meanwhile, the encoder produces a series of encoded vectors, depending on the number of words in the input sentence. Training is carried out using N epochs, with the initial epochs utilizing gold labels and the subsequent ones supporting scheduled sampling [35]. While training, the training probability p was scheduled, and a lower probability was used to encourage the model. To enhance the accuracy, label smoothing [36] was applied during the calculation of cross-entropy loss. In addition, temporal smoothing [37] was combined with a distribution probability mass to generate the neighbouring token unit in the transcript for label smoothing.

The decoder employs the attention mechanism [38, 39], which takes in the encoder output and its hidden state to compute the current hidden states $v\_1, v\_2, \ldots, v\_n$. This mechanism uses the states $s\_1, s\_2, \ldots, s\_n$ as queries and the hidden states $h\_(1:n)$ of the encoder sides as the keys and values. The attention mechanism then returns the final hidden representation, which is a weighted sum of the values.

3.5.2. CNN-LSTM Using MFCC. The CNN-LSTM technique, illustrated in Figure 2, was employed in this research. As discussed in Section 3.5.1, the method is based on MFCC feature extraction with 13, 25, and 39 features, as described in Section 3.3.1, and utilizes the CMU Sphinx language model, as outlined in Section 3.4.

3.5.3. CNN-LSTM Using Filter Bank. The CNN-LSTM method used in this research is depicted in Figure 3 and explained in Section 3.5.1. The technique applies the filter bank feature extraction approach, which projects features into a higher-dimensional space for classification enhancement [31]. The current representation utilizes a filter bank of 40, 80, and 120 coefficients to assess its recognition performance, as explained in Section 3.3.2. Furthermore, the approach incorporates the CMU Sphinx language model, as outlined in Section 3.4.

3.5.4. Sphinx-4 Tool Recognizer. A significant portion of the CMU Sphinx project, a group of speech recognition algorithms developed at Carnegie Mellon University [40], has been made available as open-source packages. The project encompasses CMU Sphinx I and II, and the development and training of CMU Sphinx III and CMU Sphinx-4 were facilitated by the CMU Sphinx train. The Sphinx train serves as the acoustic training environment for the Sphinx family of recognition engines, including Sphinx-2, Sphinx-3, and Sphinx-4. It consists of a collection of scripts, programs, and documentation aimed at creating acoustic models from data.

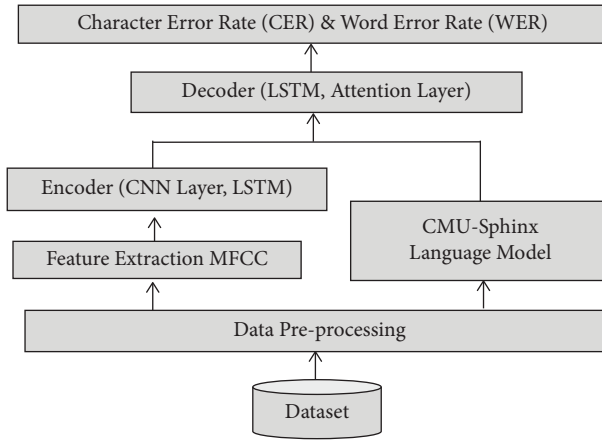

Figure 2: CNN-LSTM for automatic Arabic speech recognition using MFCC.

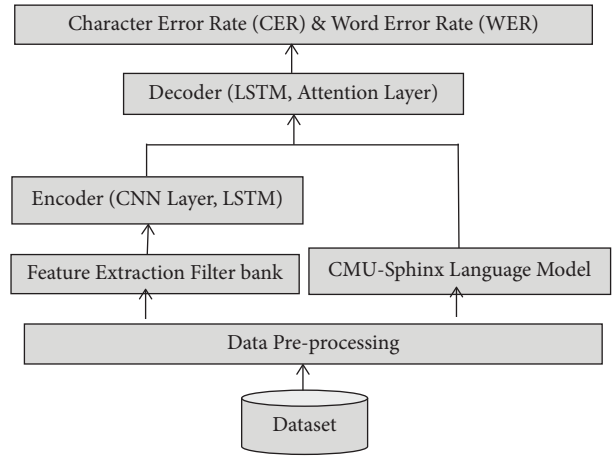

Figure 3: CNN-LSTM for automatic Arabic speech recognition using filter bank.

By converting continuous speech waveforms into a sequence of evenly spaced, discrete parameter vectors, individuals can develop models capable of recognizing the symbolic sequence of spoken syllables. The Sphinx train tool must be employed to create this model, provided there is enough acoustic data for the language and condition at hand. The Sphinx-4 architecture is designed with flexibility and adaptability in mind. It is a collection of tools for creating acoustic models from data from the Sphinx recognition engine, including algorithms, scripts, and documentation. The authors should be able to build models for any language and condition with sufficient acoustic data through this contribution. Recognition cannot proceed without an acoustic model, which is necessary to compare the data from the front end. The Sphinx train tool must be used to create this model. The final recognition result is generated by Sphinx-4 by obtaining the most probable word sequence through the result parser and subsequently outputting it [41]. This section aims to develop a structure that can recognize continuous Arabic speech using the tools provided by Carnegie Mellon University Sphinx, as illustrated in Figure 4.

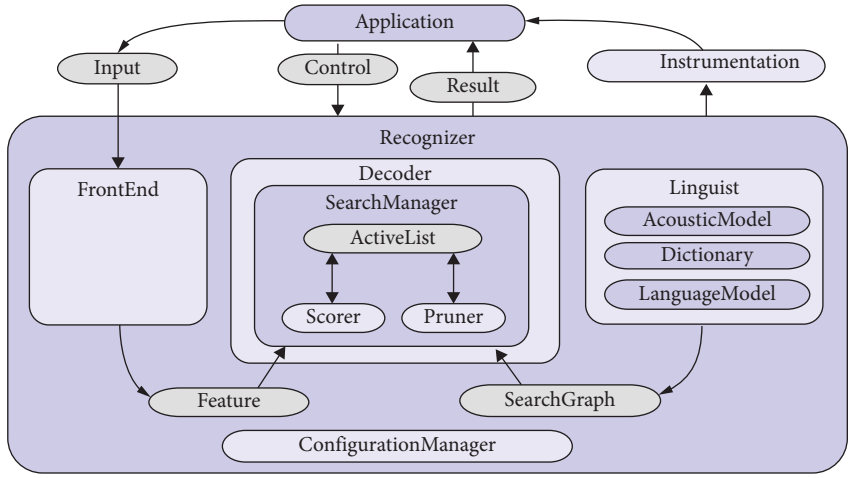

Figure 4: Carnegie Mellon University (CMU) sphinx [40].

(1) Sphinx-4 Front End. The process of speech acquisition begins with the loading of an audio file into our algorithm, which contains a sound pressure wave that forms an acoustic signal. Every speech has its unique features, and this research utilized Mel frequency cepstral coefficients (MFCC) to extract these features. The Mel-cepstrum technique is the most commonly used feature in speaker recognition. It is based on the auditory algorithm and has a high discriminating power at low frequencies compared to high frequencies. The cepstrum coefficient is capable of representing vocal tract changes dealing with convolution channel distortions and is highly resistant to noise [42].

(2) Sphinx-4 Linguist. The acoustic model is created by the linguist, which is a representation of the relationship between an acoustic signal and the phonemes (or other linguistic building blocks) that composes speech. This model is trained on a collection of audio recordings and associated transcriptions, as well as a phonetic dictionary constructed using a rule-based model. In addition, the language model assigns a probabilistic value to the whole sequence and includes a probability distribution for word sequences. By combining the knowledge from the three components, a linguist can generate a search graph suitable for the given task [43].

(3) Sphinx-4 Decoder. The search manager is generated by the decoder and is responsible for creating the scorecard, the pruner, and the active list. The scorecard is then asked to compare each token in the list to the next feature vector received from the front end, resulting in a score for each active route. The pruner uses specific heuristics to reduce the number of tokens or active routes. The result of the application is the highest-ranking path. The Sphinx-4 architecture, depicted in Figure 5, consists of three primary components: frontend, decoder, and linguist are utilized in this research. The components are based on pocket Sphinx, Sphinx base, and Sphinx train. The architecture is applied to Gaussian mixture distributions (2, 4, 8, 16, and 32), utilizing MFCC feature extraction with 13, 25, and 39 features, as illustrated in Section 3.3.1. In addition, the CMU sphinx language model is utilized, as described in Section 3.4.

4. Experimental Results and Discussion

The present research provides findings for three techniques: the initial technique employs CNN-LSTM with MFCC, the second technique employs CNN-LSTM with filter bank, and the third technique employs Sphinx-4 recognizer. All of these techniques use the continuous Arabic speech dataset and rely on the CMU Sphinx language model.

4.1. Datasets. The proposed paper used different datasets as shown in the next sections.

4.1.1. First Dataset. The first speech corpus dataset (DS1) used consists of 415 recorded Arabic sentences. To construct the corpus, 367 carefully crafted sentences with phonetic richness and balance were developed by KACST [44] and used for training the acoustic model. An additional set of 48 sentences, representing Arabic proverbs, was created by an expert in Arabic language for testing the acoustic model. The recordings for the speech corpus were performed by a total of 40 native Arabic speakers, including 20 males and 20 females, representing 11 different Arab countries. These speakers were selected to cover three major regions in the Arab world: the Levant, the Gulf, and Africa. The recording process took place in a soundproof studio using Sound Forge 8.0 software. It spanned approximately 3 months, starting from March 2009 until June 2009. The speech corpus was intentionally diversified to encompass various characteristics of Arabic native speakers. This diversification considered factors such as different age groups, nationalities, Arab regions, professions, academic qualifications, and levels of Arabic mastery. The comprehensive nature of this speech corpus, with its wide representation of Arabic speakers and attention to various characteristics, provides a valuable resource for training and evaluating ASR systems. It ensures

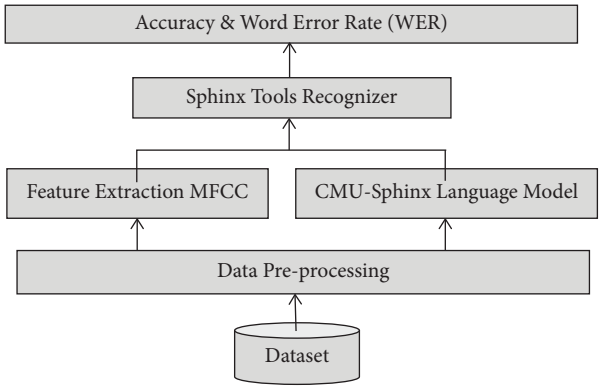

Figure 5: Sphinx-4 architecture for automatic Arabic speech recognition.

the robustness and effectiveness of the developed acoustic model across different Arabic dialects, regions, and speaker attributes.

4.1.2. Second Dataset. The second dataset (DS2) [45] features recordings of a male speaker with a Damascian accent in South Levantine Arabic. The recordings were conducted in a professional studio using a Neumann TLM 103 Studio microphone. The transcript for this corpus was sourced from the language learning website "Al Jazeera Learn," specifically chosen for its fully diacritised text, which facilitates phonetisation. To aid the speaker during recording sessions, the transcript was divided into utterances based on punctuation. The corpus has been successful in generating high-quality, natural-sounding synthesized speech. It comprises 759 utterances, totalling 2.1 hours of normal speech. This package, version 2.0 of the corpus, includes 759.wav files of spoken utterances, corresponding text utterances, and phonetic transcriptions in separate text files, following the format "[wav_filename]" "[Phoneme Sequence]" on each line.

4.2. Performance Evaluation. The proposed ASR algorithms are evaluated for their performance using various indicators, including the percentage of word error rate (WER), which is represented by equation (1); the percentage of character error rate (CER), represented by equation (2); and the percentage of accuracy, described in equation (3) [46].

$$\text{WER} = 100\%\text{-percentage of word accuracy}, \tag{1}$$

$$\text{CER} = 100\%\text{-percentage of character accuracy}, \tag{2}$$

$$\text{Percent accuracy} = \frac{N - D - S - I}{N} \times 100. \tag{3}$$

The numerical value (N) indicates the number of correctly identified labels in reference transcription; the numerical value (D) indicates the numerical value of errors caused by deletion. The numerical value (S) represents the numerical value of the substitution error, and the numerical value (I) is the numerical value of the error caused by insertion.

4.3. CNN-LSTM Using MFCC Results. Table 2 shows the results of a meticulously conducted experiment that thoroughly examined the performance of a CNN-LSTM model using MFCC across multiple datasets. The first dataset is presented in detail in Section 4.1.1, and the second dataset is elaborated upon in Section 4.1.2. The examination employed WER and CER metrics. The model comprises a hybrid CNN-LSTM for encoding and an attention-based decoding model, while the language model used CMU Sphinx with 17236 uni-grams, 42659 bi-grams, and 50147 tri-grams. The experiment also involved MFCC feature extraction with varying numbers of features, specifically 13, 25, and 39.

The findings presented in Table 2 unequivocally demonstrate that the utilization of MFCC 25 features yields superior results in terms of word error rate in DS2 compared to DS1 across all features of MFCC. Moreover, the evaluation of the character error rate reveals that the performance of MFCC 25 features in DS2 surpasses that of MFCC 25 features in DS1, once again across all features of MFCC. These results robustly highlight the consistent advantages offered by MFCC 25 features in both word and character error rates, underscoring their effectiveness and reliability in speech-processing applications.

4.4. CNN-LSTM Using Filter Bank Results. Table 3 displays the experimental findings, presenting an evaluation of the performance of a CNN-LSTM model that employs filter banks across various datasets. The first dataset, extensively discussed in Section 4.1.1, and the second dataset, elaborated on in Section 4.1.2, are both examined in detail to provide a comprehensive understanding of the results. The evaluation was created using data from WER and CER. The model consists of a hybrid CNN-LSTM for encoding and an attention-based decoding model, while the language model used CMU Sphinx with 17236 uni-grams, 42659 bi-grams, and 50147 tri-grams. The experiment also involved filter bank feature extraction with varying numbers of features, specifically 40, 80, and 120, as presented in the table.

The analysis of the outcomes presented in Table 3 unambiguously demonstrates the superiority of filter bank 120 features in terms of word error rate in DS2 compared to DS1 across all features of filter bank. In addition, the evaluation of

Table 2: WER and CER results of CNN-LSTM using MFCC.

No. of MFCC features	WER		CER	
	DS1 (%)	DS2 (%)	DS1 (%)	DS2 (%)
13 features	4.21	2.26	4.18	3.61
25 features	4.04	**1.26**	3.83	**2.86**
39 features	4.04	1.43	3.82	3.36

The bold values indicate the best results of CNN-LSTM using MFCC from WER and CER.

Table 3: WER and CER results of CNN-LSTM using filter.

No. of filter bank features	WER		CER	
	DS1 (%)	DS2 (%)	DS1 (%)	DS2 (%)
40 features	4.21	4.21	4.42	2.98
80 features	4.50	3.54	4.12	**1.27**
120 features	3.63	**3.46**	4.05	1.82

The bold values indicate the best results of CNN-LSTM using filter bank from WER and CER.

Table 4: WER and accuracy results of Sphinx-4 recognizer.

MFCC features	Gaussian 2		Gaussian 4		Gaussian 8		Gaussian 16		Gaussian 32	
	WER (%)	Acc. (%)	WER (%)	Acc. (%)	WER (%)	Acc. (%)	WER (%)	Acc. (%)	WER (%)	Acc. (%)
13 features	13.39	86.61	11.02	88.98	11.13	88.87	10.85	89.15	14.25	85.75
25 features	10.77	89.23	8.45	91.55	**8.17**	**91.83**	8.65	91.35	10.85	89.15
39 features	14.70	85.30	12.78	87.22	10.91	89.09	10.77	89.23	13.86	86.14
52 features	23.60	76.56	20.81	79.19	18.41	81.59	18.33	81.67	32.01	76.99

The bold values indicate the best results of Sphinx-4 recognizer from WER and accuracy.

the character error rate reveals that the performance of filter bank 80 features in DS2 outperforms that of DS1, once again across all features of the filter bank. These compelling results emphasize the consistent advantages offered by filter bank 120 features in word error rate, as well as the effectiveness of filter bank 80 features in character error rate. Together, these findings underscore the robust performance and reliability of filter bank features across a range of speech-processing applications.

4.5. Sphinx-4 Recognizer Results. In this experiment, a variety of MFCC feature methods were tested using Sphinx-4. These methods included MFCC_0 with 13 features, MFCC_D with 25 features, MFCC_D_A with 39 features, and MFCC_E_D_A_Z with 52 features. The experiment also employed a CMU Sphinx language model based on 17236 uni-grams, 42659 bi-grams, and 50147 tri-grams. Gaussian mixture distributions with different numbers (2, 4, 8, 16, and 32) were used in conjunction with the first dataset [44].

A thorough examination of the outcomes presented in Table 4 unequivocally demonstrates the exceptional performance of MFCC 25 features and Gaussian 8 in terms of word error rate and accuracy when evaluated based on DS1. Notably, across all features of MFCC and Gaussian degrees, the results consistently indicate that MFCC 25 features and Gaussian 8 exhibit the highest level of efficacy. These compelling findings emphasize the robustness and reliability of utilizing MFCC 25 features and Gaussian 8, highlighting their significant impact on improving word error rate and accuracy in various speech-processing applications.

4.6. Statistical Analysis for the Results in the Proposed Paper. Table 5 and Figure 6 show conducting two experiments on two distinct datasets, namely, the large dataset DS1 and the small dataset DS2, utilizing the CNN-LSTM technique. Our findings revealed crucial insights. Specifically, we observed that the filter bank 120 features exhibited significant advantages and yielded higher proficiency when applied to the large dataset DS1 which has 3.63% WER as a standard comparison measurement. Conversely, the MFCC 25 features demonstrated superior performance and greater proficiency when employed on the small dataset DS2, which has 1.26% WER. These insightful conclusions highlight the importance of tailoring feature selection based on dataset characteristics, thereby maximizing the effectiveness and proficiency of the chosen techniques in different scenarios.

By conducting an extensive comparative analysis between the current paper and the work presented by Abushariah et al. [47], a meticulous examination of the performance criteria was undertaken. Notably, Table 6 serves as a valuable resource, presenting comprehensive insights into the word error ate (WER) metric as a standard comparison measurement and shedding light on the effectiveness of the selected approach. These insightful findings were obtained through the utilization of DS1, wherein Gaussian mixture distributions were carefully configured to 2, 8, 16, and 32. Furthermore, the Sphinx-4 model was employed as the designated speech recognition engine, ensuring the reliability, consistency, and robustness of the obtained results.

Table 5: Comparison of WER results based on different datasets using the CNN-LSTM technique.

Features	CNN-LSTM technique	
	DS1 (%)	DS2 (%)
MFCC 13 features	4.21	2.26
MFCC 25 features	4.04	**1.26**
MFCC 39 features	4.04	1.43
Filter bank 40 features	4.21	4.21
Filter bank 80 features	4.50	3.54
Filter bank 120 features	**3.63**	3.46

The bold values indicate the best comparison of WER results based on different datasets using the CNN-LSTM technique.

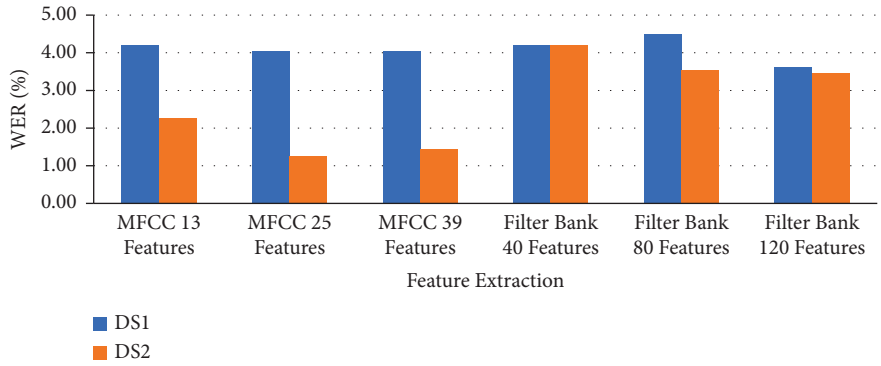

Figure 6: Comparison of WER results based on different datasets using the CNN-LSTM technique.

Table 6: Comparative WER results between this research and Abushariah et al. [47].

Gaussian mixture distributions	Sphinx-4 recognizer	
	This research (%)	Abushariah et al. [47]
Gaussian 2	10.77	—
Gaussian 4	8.45	—
Gaussian 8	8.17	—
Gaussian 16	8.65	14.44%
Gaussian 32	10.85	—

By conducting a meticulous analysis of the results, it was observed that Gaussian 8 yielded the best outcome in this research by using the Sphinx-4 recognizer. Furthermore, upon comparing the results of this research with those of Abushariah et al. [47], it became evident that this research's implementation of Gaussian 16 extracted superior results compared to the work by Abushariah et al. [47], which has 8.65% WER compared to 14.44% of Abushariah et al. [47]. These comparative findings highlight the advancements and improvements achieved in this research, particularly in terms of Gaussian mixture distributions, emphasizing the enhanced performance and potential for more accurate speech recognition.

5. Conclusion and Future Work

This study aimed to develop a continuous Arabic speech recognition algorithm using two techniques. The first technique involves a hybrid encoding model consisting of CNN and LSTM, and a decoding model based on attention. The second technique is Sphinx-4, which uses two types of feature extraction (MFCC and filter bank) and a language model that includes uni-grams, bi-grams, and tri-grams created using CMU Sphinx. The experiment involved analysing 6139 WAV files from different speakers, totalling 7 hours of recording. The CNN-LSTM technique achieved a WER of 3.63% using 120 features for filter bank and 4.04% WER using 39 features for MFCC, while the Sphinx-4 recognizer achieved a WER of 8.17% and an accuracy of 91.83% using 25 features for MFCC and 8 Gaussian mixtures. Our future research will involve examining a combined CNN-LSTM model and an end-to-end technique that uses attention to other vast datasets. Furthermore, we propose enhancing the external language model to improve the accuracy and effectiveness of the models, reducing the word error rate.

Data Availability

The first dataset that used in this article is included in the article [44] and the second dataset is included in the article [45].

Conflicts of Interest

All authors declare that they have no potential conflicts of interest.

References

[1] Z. J. Mohammed Ameen and A. Abdulrahman Kadhim, "Deep learning methods for Arabic autoencoder speech recognition system for electro-larynx device," *Advances in*

Human-Computer Interaction, vol. 2023, Article ID 7398538, 11 pages, 2023.

[2] E. Alsharhan and A. Ramsay, "Investigating the effects of gender, dialect, and training size on the performance of Arabic speech recognition," *Language Resources and Evaluation*, vol. 54, no. 4, pp. 975–998, 2020.

[3] D. Vergyri, K. Kirchhoff, K. Duh, and A. Stolcke, *Morphology-based Language Modeling for Arabic Speech Recognition*, SRI International Menlo Park United States, Menlo Park, CA, USA, 2004.

[4] H. Satori, H. Hiyassat, M. Haiti, and N. Chenfour, "Investigation Arabic speech recognition using CMU sphinx system," *The International Arab Journal of Information Technology*, vol. 6, no. 2, 2009.

[5] A. Ali, H. Mubarak, and S. Vogel, "Advances in dialectal Arabic speech recognition: a study using twitter to improve egyptian asr," in *Proceedings of the 11th International Workshop on Spoken Language Translation: Papers*, pp. 156–162, Doha, Qatar, December 2014.

[6] P. Cardinal, A. Ali, N. Dehak et al., "Recent advances in ASR applied to an Arabic transcription system for Al-Jazeera," in *Proceedings of the Fifteenth annual conference of the international speech communication association*, Singapore, September 2014.

[7] F. Diehl, M. J. Gales, M. Tomalin, and P. C. Woodland, "Morphological decomposition in Arabic ASR systems," *Computer Speech & Language*, vol. 26, no. 4, pp. 229–243, 2012.

[8] O. Hamed and T. Zesch, "A survey and comparative study of Arabic diacritization tools," *Journal for Language Technology and Computational Linguistics*, vol. 32, no. 1, pp. 27–47, 2017.

[9] I. Hadjir, M. Abbache, and F. Z. Belkredim, "An approach for Arabic diacritization," in *Natural Language Processing and Information Systems: 24th International Conference on Applications of Natural Language to Information Systems, NLDB 2019*, pp. 337–344, Springer, Salford, UK, 2019.

[10] M. A. Hassan, A. Rehmat, M. U. Ghani Khan, and M. H. Yousaf, "Improvement in automatic speech recognition of south asian accent using transfer learning of deepspeech2," *Mathematical Problems in Engineering*, vol. 2022, Article ID 6825555, 12 pages, 2022.

[11] Y. Belinkov, A. Ali, and J. Glass, "Analyzing phonetic and graphemic representations in end-to-end automatic speech recognition," 2019, https://arxiv.org/abs/1907.04224.

[12] T. Nagamine, M. L. Seltzer, and N. Mesgarani, "Exploring how deep neural networks form phonemic categories," in *Proceedings of the Sixteenth Annual Conference of the International Speech Communication Association*, Dresden, Germany, September 2015.

[13] T. Nagamine, M. L. Seltzer, and N. Mesgarani, *On the Role of Nonlinear Transformations in Deep Neural Network Acoustic Models*, Interspeech, Sydney, Australia, 2016.

[14] T. Hori, J. Cho, and S. Watanabe, "End-to-end speech recognition with word-based RNN language models," in *Proceedings of the 2018 IEEE Spoken Language Technology Workshop (SLT)*, pp. 389–396, IEEE, Athens, Greece, December 2018.

[15] P. Motlicek, D. Imseng, B. Potard, P. N. Garner, and I. Himawan, "Exploiting foreign resources for DNN-based ASR," *EURASIP Journal on Audio Speech and Music Processing*, vol. 2015, pp. 17–10, 2015.

[16] M. Attia, Y. Samih, K. Shaalan, and J. Van Genabith, "The floating Arabic dictionary: an automatic method for updating a lexical database through the detection and lemmatization of unknown words," in *Proceedings of COLING 2012*, pp. 83–96, Mumbai, India, December 2012.

[17] A. Ahmed, Y. Hifny, K. Shaalan, and S. Toral, "End-to-end lexicon free Arabic speech recognition using recurrent neural networks," in *Computational Linguistics, Speech and Image Processing for Arabic Language*, pp. 231–248, World Scientific, Singapore, 2019.

[18] K. Khatatneh, "A novel Arabic Speech Recognition method using neural networks and Gaussian Filtering," *International Journal of Electrical, Electronics and Computer Systems*, vol. 19, no. 1, 2014.

[19] A. E.-D. Mousa, H.-K. J. Kuo, L. Mangu, and H. Soltau, "Morpheme-based feature-rich language models using deep neural networks for lvcsr of Egyptian Arabic," in *Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8435–8439, IEEE, Vancouver, BC, Canada, May 2013.

[20] T. AlHanai, W.-N. Hsu, and J. Glass, "Development of the MIT ASR system for the 2016 Arabic multi-genre broadcast challenge," in *Proceedings of the 2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 299–304, IEEE, San Diego, CA, USA, December 2016.

[21] A. Ali, Y. Zhang, P. Cardinal, N. Dahak, S. Vogel, and J. Glass, "A complete KALDI recipe for building Arabic speech recognition systems," in *Proceedings of the 2014 IEEE spoken language technology workshop (SLT)*, pp. 525–529, IEEE, South Lake Tahoe, NV, USA, December 2014.

[22] A. AbdAlmisreb, A. F. Abidin, and N. M. Tahir, "Maxout based deep neural networks for Arabic phonemes recognition," in *Proceedings of the 2015 IEEE 11th International Colloquium on Signal Processing & its Applications (CSPA)*, pp. 192–197, IEEE, Kuala Lumpur, Malaysia, March 2015.

[23] M. A Azim, A. Aziza Hamid, N. L Badr, and M. Tolba, "Large vocabulary Arabic continuous speech recognition using tied states acoustic models," *Asian Journal of Information Technology*, vol. 18, no. 2, pp. 49–56, 2019.

[24] H. A. Alsayadi, A. A. Abdelhamid, I. Hegazy, and Z. T. Fayed, "Arabic speech recognition using end-to-end deep learning," *IET Signal Processing*, vol. 15, no. 8, pp. 521–534, 2021.

[25] T. Kudo and J. Richardson, "Sentencepiece: a simple and language independent subword tokenizer and detokenizer for neural text processing," *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2018, https://arxiv.org/abs/1808.06226.

[26] J. G. Roederer, "The perception of music by the human brain: an introductory course," *Journal of the Acoustical Society of America*, vol. 105, no. 2, p. 1054, 1999.

[27] T. S. Shanthi and C. Lingam, "Review of feature extraction techniques in automatic speech recognition," *International Journal of Scientific Engineering and Technology*, vol. 2, no. 6, pp. 479–484, 2013.

[28] Y. Xu, "English speech recognition and evaluation of pronunciation quality using deep learning," *Mobile Information Systems*, vol. 2022, Article ID 7186375, 12 pages, 2022.

[29] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: a survey," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 9411–9457, 2021.

[30] Y. He, Y. Liu, S. Shao et al., "Application of CNN-LSTM in gradual changing fault diagnosis of rod pumping system," *Mathematical Problems in Engineering*, vol. 2019, Article ID 4203821, 9 pages, 2019.

[31] J. Chorowski, D. Bahdanau, K. Cho, and Y. Bengio, "End-to-end continuous speech recognition using attention-based

recurrent nn: first results," 2014, https://arxiv.org/abs/1412.1602.

[32] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proceedings of the 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4845–4849, IEEE, New Orleans, LA, USA, March 2017.

[33] S. Watanabe, T. Hori, S. Karita et al., "Espnet: end-to-end speech processing toolkit," 2018, https://arxiv.org/abs/1804.00015.

[34] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM networks for improved phoneme classification and recognition," in *Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005: 15th International Conference*, pp. 799–804, Springer, Warsaw, Poland, 2005.

[35] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[36] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.

[37] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," 2016, https://arxiv.org/abs/1612.02695.

[38] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, https://arxiv.org/abs/1409.0473.

[39] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, https://arxiv.org/abs/1508.04025.

[40] H. Satori, M. Harti, and N. Chenfour, "Arabic speech recognition system using cmu-sphinx4," 2007, https://arxiv.org/abs/0704.2201.

[41] X. Wang, "Research on open oral English scoring system based on neural network," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 1346543, 9 pages, 2022.

[42] P. Lamere, "The CMU SPHINX-4 speech recognition system," *IEEE International Conference on acoustics, speech and signal processing (icassp 2003), hong kong*, vol. 1, pp. 2–5, 2003.

[43] V. Këpuska and G. Bohouta, "Comparing speech recognition systems (microsoft API, google API and CMU sphinx)," *International Journal of Engineering Research in Africa*, vol. 07, no. 3, pp. 20–24, 2017.

[44] M. Alghamdi, A. Alhamid, and M. Aldasuqi, "Database of Arabic sounds: sentences," Technical Report, King Abdulaziz City of Science and Technology, Riyadh, Saudi Arabia, 2003.

[45] N. Halabi and M. Wald, "Phonetic inventory for an Arabic speech corpus," 2016, https://eprints.soton.ac.uk/397310/1/Arabic%2520Phonetic%2520Vocab%25202016.pdf.

[46] B. A. Al-Qatab and R. N. Ainon, "Arabic speech recognition using hidden Markov model toolkit (HTK)," *2010 international symposium on information technology*, vol. 2, pp. 557–562, 2010.

[47] M. A.-A. M. Abushariah, R. N. Ainon, R. Zainuddin, M. Elshafei, and O. O. Khalifa, "Arabic speaker-independent continuous automatic speech recognition based on a phonetically rich and balanced speech corpus," *The International Arab Journal of Information Technology*, vol. 9, no. 1, pp. 84–93, 2012.