Correct pronunciation detection of the arabic alphabet using deep learning

Ziafat, Nishmia; Ahmad, Hafiz Farooq; Fatima, Iram; Zia, Muhammad; Alhumam, Abdulaziz; Rajpoot, Kashif

DOI:
10.3390/app11062508

applied sciences

MDPI

Article

Correct Pronunciation Detection of the Arabic Alphabet Using Deep Learning

Nishmia Ziafat [1,*], Hafiz Farooq Ahmad [2], Iram Fatima [2], Muhammad Zia [1], Abdulaziz Alhumam [2] and Kashif Rajpoot [3,4]

[1] COMSIP Lab, Department of Electronics, Quaid-I-Azam University, Islamabad 45320, Pakistan; ziaasghar@gmail.com
[2] Computer Science Department, College of Computer Sciences and Information Technology (CCSIT), King Faisal University, P.O. Box 400, Al-Ahsa 31982, Saudi Arabia; hfahmad@kfu.edu.sa (H.F.A.); iram.fa@gmail.com (I.F.); aahumam@kfu.edu.sa (A.A.)
[3] School of Electrical Engineering and Computer Science, NUST, Islamabad 44000, Pakistan; kashif.rajpoot@oxfordalumni.org
[4] School of Computer Science, University of Birmingham, Edgbaston, Birmingham B15 2TT, UK
* Correspondence: nziafat@ele.qau.pk

Abstract: Automatic speech recognition for Arabic has its unique challenges and there has been relatively slow progress in this domain. Specifically, Classic Arabic has received even less research attention. The correct pronunciation of the Arabic alphabet has significant implications on the meaning of words. In this work, we have designed learning models for the Arabic alphabet classification based on the correct pronunciation of an alphabet. The correct pronunciation classification of the Arabic alphabet is a challenging task for the research community. We divide the problem into two steps, firstly we train the model to recognize an alphabet, namely Arabic alphabet classification. Secondly, we train the model to determine its quality of pronunciation, namely Arabic alphabet pronunciation classification. Due to the less availability of audio data of this kind, we had to collect audio data from the experts, and novices for our model's training. To train these models, we extract pronunciation features from audio data of the Arabic alphabet using mel-spectrogram. We have employed a deep convolution neural network (DCNN), AlexNet with transfer learning, and bidirectional long short-term memory (BLSTM), a type of recurrent neural network (RNN), for the classification of the audio data. For alphabet classification, DCNN, AlexNet, and BLSTM achieve an accuracy of 95.95%, 98.41%, and 88.32%, respectively. For Arabic alphabet pronunciation classification, DCNN, AlexNet, and BLSTM achieve an accuracy of 97.88%, 99.14%, and 77.71%, respectively.

Keywords: deep learning (DL); artificial neural network (ANN); deep convolution neural network (DCNN); recurrent neural network (RNN); bidirectional long short-term memory (BLSTM)

Citation: Ziafat, N.; Ahmad, H.F.; Fatima, I.; Zia, M.; Alhumam, A.; Rajpoot, K. Correct Pronunciation Detection of the Arabic Alphabet Using Deep Learning. *Appl. Sci.* **2021**, *11*, 2508. https://doi.org/10.3390/app11062508

Academic Editor: Byung-Gyu Kim

Received: 14 January 2021
Accepted: 5 March 2021
Published: 11 March 2021

1. Introduction

The Arabic language is one of the oldest languages and is characterized due to its uniqueness and flexibility. Among many Semitic languages, Arabic is the most widely spoken language with over 290 million native speakers and 132 million non-native speakers [1]. Arabic is one of the six official languages of the United Nations (UN) [2]. Classical Arabic (CA) and modern standard Arabic (MSA) are the two main dialects of Arabic. CA is the language of the Quran while MSA is its modified version, which is currently used in everyday communication.

Rules of pronunciation are very well-defined for CA to preserve the accurate meaning of the words and constitute basic building blocks to help natives as well as non-natives to learn the Arabic language. The requirements to consider for correct pronunciation are the articulation points of the alphabets, characteristics of the alphabets, and extensive practicing of vocals [3]. This research work focuses on developing an automated system that can

recognize the correct pronunciation of the Arabic alphabet. This research is an important milestone for developing and classify a more sophisticated system that can automatically classify words and sentences to help in teaching classical Arabic pronunciation.

In this research, we take users' audio data, process, and train neural networks (NN) over this data. The network learns from the data and classifies audio data and hence provides feedback to a learner on the alphabet pronunciation.

Automatic speech recognition (ASR) [4] has received considerable attention and recently made significant progress with its applications in mobile computing [5], human–computer interaction [6,7], information retrieval [8], and assisted communication [9]. ASR is a process of recognizing information from spoken words. Generally, ASR algorithms use acoustic, pronunciation, and language modeling [4,10]. ASR has active research attention in different human languages [11]. In addition to ASR, many studies performed speech recognition with mispronunciation detection for children and other non-native language learners in many languages, i.e., English [12–14], and Mandarin Chinese [15,16]. However, limited work has been done in Arabic ASR using pattern recognition and feature extraction techniques. Pattern recognition techniques used in ASR incorporate hidden Markov model (HMM) [4,17,18], Gaussian mixture model (GMM) [4,19], artificial neural network (ANN) [20], and multi-layer perceptron (MLP) [20] using different feature extraction techniques such as mel-frequency cepstral coefficient (MFCC), linear predictive cepstrum coefficients (LPCC) [21], and spectrogram.

The HMM [17,18] determines the set of states and associates them with the probabilities of transitions between these states called the Markov chain. GMM [19] is a probabilistic model, which represents a normally distributed subclass within a class. Mixture models do not know a data point belonging to a subclass, and it allows the model to learn automatically. During the past decade, a few HMM and NN speech recognition systems have demonstrated to provide higher accuracy in the classical Arabic alphabet and verse recognition tasks. CMU Sphinx is one of the well-known open-source tools for CA based on HMM [22–25]. Researchers worked on different tasks using HMM such as an 'E-Hafiz system' which was proposed for CA learning using HMM and MFCC as a feature learning technique [26]. This system achieved an accuracy of 92% for men and 90% for children. In [27], the proposed system helps to improve the pronunciation of alphabets using mean square error (MSE) for pattern matching and MFCC as a feature extraction technique. This system successfully recognized correct pronunciation for various alphabets. In [28], the 'Tajweed checking system' demonstrates detection and correction of students' mistakes during recitations using MFCC, and vector quantization (VQ) with an accuracy of 82.1–95%. In [29], a 'Qalqalah letter pronunciation' is proposed using spectrogram, this technique illustrates the mechanism of Qalqalah sound pronunciation. In [30], a 'mispronunciation detection system for Qalqalah letters' is proposed using the MFCC, and support vector machine (SVM) classifier, which provides an accuracy of 97.5%.

Deep learning (DL) algorithms learn a hierarchical representation from data with numerous layers [31]. The hidden layers are responsible to extract important features from the raw data to achieve a better representation of the audio data. In [32], the author proposes an Arabic alphabet recognition model using RNN with back-propagation through time, with an accuracy of 82.3% tested for 20 alphabets. In recent research [33], the authors demonstrated a mispronunciation detection system using different handcrafted techniques for feature extraction and SVM, KNN, and NN as classifiers. This experiment achieved an accuracy of 74.37% for KNN, 83.90% for SVM, and 90.1% for NN.

In this paper, we propose a DL algorithm, i.e., DCNN, AlexNet, and BLSTM neural networks, for the Arabic alphabet classification. Our research is different from the previous works in terms of the dataset, features extraction, network architecture, and performance. We employ mel-spectrogram for extracting features of the audio dataset of alphabets. The mel-spectrogram is the conversion of audio frames into frequency-domain representation, which are scaled on an equally spaced mel-scale. The magnitude or power spectrum passes through the mel-filter to obtain the mel-spectrogram. The previous approaches

mostly use MFCC, which is related to mel-spectrogram. MFCC coefficients are obtained by passing a mel-spectrum through a logarithmic scale and then discrete cosine transform (DCT). Due to excessive use of DL in speech systems, DCT is no longer a necessary step [34].

We are working on two classification problems using an audio dataset of the Arabic alphabet. The first problem is a multi-class classification task to detect and classify the alphabet to their respective classes. The second problem is a binary class classification task that detects and classifies the correct and incorrect pronunciation to their respective class. The models we use in this research are CNN model learns features from mel-spectrogram and BLSTM learns to use the spectral features technique. In this paper, our major contributions are:

1. Collection of an audio dataset for the Arabic alphabet with correct and mispronunciation.
2. Arabic alphabet classification (recognize each alphabet).
3. Arabic alphabet pronunciation classification (detect correct pronunciation of the alphabet).
4. Exploration of DCNN, AlexNet, and BLSTM to perform classification of the audio set of the Arabic alphabet.

The Arabic language has 29 alphabets, and we consider each alphabet as a class. The Arabic alphabet classification is a multi-class classification problem, which involves the Arabic alphabet audio dataset, and the classification task recognizes each alphabet's class. The audio file with the alphabet sequence shown in Figure 1, is fed into NN. The network learns and extracts the feature set of each alphabet based on its characteristics. The classifier then evaluates and differentiates the alphabet in their respected class. On the other hand, Arabic alphabet pronunciation classification is a binary classification problem. In this task, our focus is to detect the correct pronunciation of the alphabet. The network learns characteristics of the dataset and classifies them into correct pronunciation and mispronunciation classes.

Letter Sequence	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Classic pronunciation	ʾalif	bāʾ	tāʾ	*sāʾ*	jeem	ḥhāʾ	khāʾ	dāl	zhāl	rāʾ	zā	seen	sheen	ṣuād	ḍuād
Arabic script	أَلِف	بَاء	تَاء	ثَاء	جِيم	حَاء	خَاء	دَال	ذَال	رَاء	زَاي	سِين	شِين	صَاد	ضَاد
IPA symbol	/aː/	/b/	/t/	/θ/	/d͡ʒ/	/ħ/	/x/	/d/	/ð/	/r/	/z/	/s/	/ʃ/	/sˤ/	/dˤ/
Isolated Form	ا	ب	ت	ث	ج	ح	خ	د	ذ	ر	ز	س	ش	ص	ض
Letter Sequence	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	
Classic pronunciation	ṭuāʾ	ẓuāʾ	ʿaain	ghain	fāʾ	qāuf	kāaf	lāam	meem	noon	hāʾ	wao	yāʾ	hamzah	
Arabic script	طَاء	ظَاء	عَيْن	غَيْن	فَاء	قَاف	كَاف	لَام	مِيم	نُون	هَاء	وَاو	يَاء	هَمْزة	
IPA symbol	/tˤ/	/ðˤ/	/ʕ/	/ɣ/	/f/	/q/	/k/	/l/	/m/	/n/	/h/	/w/	/j/, /iː/	(used as	
Isolated Form	ط	ظ	ع	غ	ف	ق	ك	ل	م	ن	ه	و	ي	ء	

Figure 1. Arabic alphabet representation in classical, Arabic script, and international phonetic alphabets.

The organization of the rest of the paper is as follows. Section 2 explains the collection and preprocessing of the data. Section 3 presents the proposed methodology and DL classification models. In Section 4, we present experimental results and their comparative analysis. Section 5 concludes our work.

2. Data Collection and Preprocessing

In this section, we present the data collection and preprocessing technique applied to the dataset. These techniques can have a significant impact on the training of the learning model [35]. The collected audio samples of the Arabic alphabet may have noise and background speech, which causes distortion and can affect the decision of the classifier [36]. The preprocessing reduces noise and background speech from the collected samples.

2.1. Noise Reduction

Several algorithms and applications are available for speech enhancement. We performed noise suppression using spectral subtraction and voice activity detection [37] over noisy audio samples. We have obtained spectral estimates for the background noise from the input signal. Figure 2 demonstrates the performance of the proposed method; Figure 2a shows the time-waveform of the alphabet with background noise whereas Figure 2b shows the time-waveform of the clean alphabet.

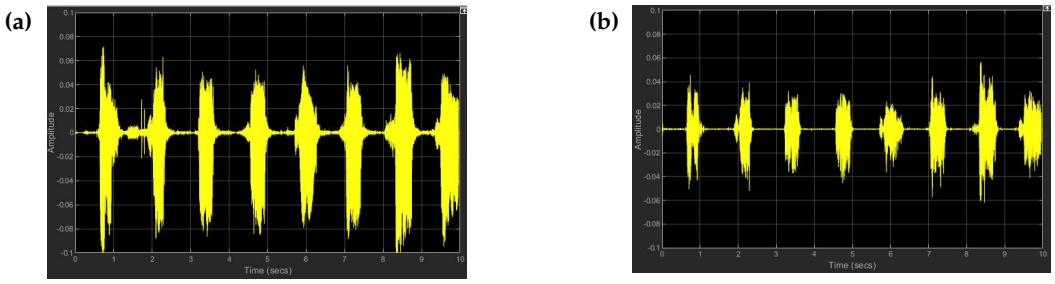


Figure 2. Noise and background speech removal from the recorded audio samples. (**a**) Time-waveform of the noisy audio samples. (**b**) Time-waveform of the clean audio samples.

2.2. Audio Segmentation and Silence Removal

Silence is an unvoiced part of a speech signal and it is useful in detecting pauses between speech but most of the time it is useless because it makes extraction of actual information difficult [36]. We adjusted each clip to have minimal silence because silence makes it difficult for the network to classify an alphabet accurately due to the presence of useless information. Recorded data files consist of 29 letters and each of them is separated through silence, which is useful in the segmentation of a large file. We implemented a speech detection algorithm over the audio dataset, the algorithm is based on [38]. The algorithm detects the boundaries of the speech and discards the silence at the end and beginning of the speech. It transforms the audio signal to time–frequency representation with specified 'Window' and 'OverlapLength' (Number of samples overlapping between adjacent windows). For each frame, it calculates short-term energy and spectral spread and then creates their histogram.

The spectral spread and short-term energy are smoothed over time by passing through the successive moving median filter to alleviate spikes that are left after noise removal and compared with their respective threshold to create the mask. The masks are combined and the speech regions are merged with 'MergeDistance' (Number of samples over which merge positive speech is detected) to declare a frame with speech. Figure 3 shows the detected speech levels discarding the silence between them. Later, we save these speech segments in separate audio files, so we can use them in an audio classification task.

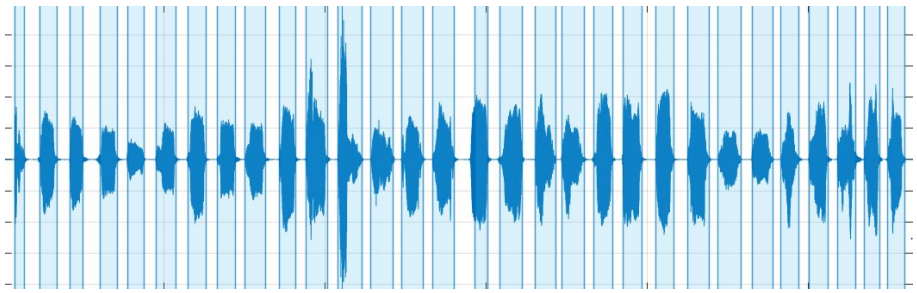
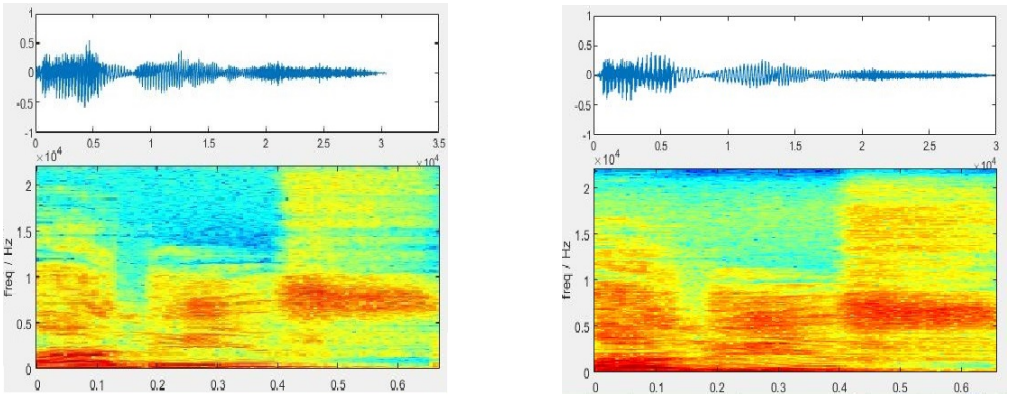


Figure 3. Audio levels are marked according to speech detected in the recorded file before splitting.

2.3. Data Augmentation

Data augmentation is a familiar ML strategy, and we use it to increase data quantity [39]. We enhanced the data by modifying the existing source data, we augmented about 20 samples of each alphabet from the existing dataset. In audio data augmentation, we used a pitch variation factor to retain the originality of the audio dataset and have minimal effect on the pronunciation. We found this technique suitable for this work after cross-checking audio files audibly. It was also the only technique that did not have any negative impact on the Arabic audio dataset. We obtained 6 modified samples from each alphabet by varying pitch between levels −0.3, and 0.3. Figure 4a shows time–frequency representation without augmentation and Figure 4b shows time–frequency representation with augmentation. This augmented dataset was cross-checked by an expert to ensure the pronunciation of the alphabet is not compromised during augmentation.



(a) **(b)**

Figure 4. Time and frequency representation of audio file: (**a**) without data augmentation, (**b**) with augmentation.

3. Methodology

This section presents the methodology and different stages of this research work. This research work consists of five stages: data collection, preprocessing, feature extraction, network training, and classification of unseen data. These stages are described through the system architecture as shown in Figure 5.

The first and second stage of this proposed methodology involves the collection of the dataset and preprocessing, we have already discussed these two stages in the previous section. The third stage involves feature extraction; the features are extracted from the raw data and input to the fourth stage for training the network using deep learning models. In the end, we compare the training data with unseen testing data. Then we estimate the accuracy and display the confusion charts for each class.

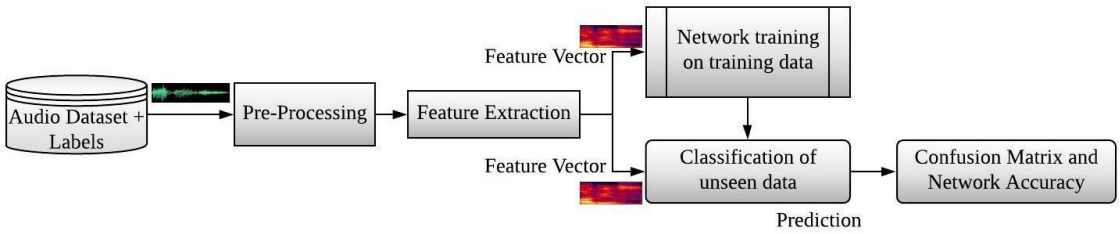


Figure 5. Proposed system architecture.

DCNN, AlexNet, and BLSTM algorithms applicable to both the Arabic alphabet classification problem and the alphabet pronunciation classification problem. The only difference is in the dataset and the number of classes. The evaluation experiment conducted in this work is speaker-open/speaker-independent.

3.1. Feature Extraction

In ASR systems, we extract a feature set from the speech signals. The classification algorithm is performed on the features set instead of speech signals directly. Feature extraction provides a compact representation of speech waveforms. Classification-based feature extraction reduces redundancy and removes the irrelevant information in large datasets [40]. A large dataset requires huge memory and computation power and leads to over-fitting.

CNN extracts feature autonomously and converts the raw audio data into mel-spectrogram [41]. We have done this conversion only for CNN (DCNN and AlexNet) as it takes an input image, processes it, and then classifies it in different categories. CNN extracts and filters an enormous number of features to get useful features for the classification of the audio alphabet. In this work, we are using filtered features, from the FC layer.

On the other hand, BLSTM needs assistance for extracting features. In the BLSTM network, we extract the information of the given dataset using spectral features from the raw audio data. The extracted data are stored and later given as input to the BLSTM network for training, testing, and evaluation of audio alphabet [42]. First, we use mel-spectrum with BLSTM, but the results were not promising, so we opted toward handcrafted features. We extract 12 spectral features from the raw data including spectral centroid, spectral spread, spectral skewness, spectral kurtosis, spectral entropy, spectral flatness, spectral crest, spectral flux, spectral slope, spectral decrease, spectral roll-off point, and pitch. These features are widely used in machine learning, deep learning applications, and perceptual analysis. We are using these features to differentiate notes, pitch, rhythm, and melody of speech.

3.2. Neural Network Model Training

The development of the learning model requires a history of the training data and provide observation of the data with input. The network captures the meaning of these observations in the output. The neural network learns a mapping function to find an optimal set of model parameters. We tested different network parameters and after their empirical analysis, the following parameter values are used as shown in Table 1.

Table 1. Training option of the neural network.

Parameters	DCNN	AlexNet	BLSTM
Learning Rate	0.0001	0.00001	0.001
Epochs	35	35	100
Batch Size	75	373	126
Optimizing Algorithm		Adam [43]	

3.3. Deep Learning Models for Classification

DL consists of vast models and several associated algorithms. The dataset and the type of tasks performed play a significant role in selecting a model. The audio alphabet dataset is trained and tested using deep learning models to achieve better accuracy and minimum loss function. The pre-trained models on the Quranic dataset are not available, so we trained the algorithms from scratch and by fine-tuning the existing models. Two types of models are selected for the classification of audio alphabets:

1. Convolution neural network (CNN)
   (a) DCNN
   (b) AlexNet (by transfer learning approach)
2. Recurrent neural network (RNN)
   (a) BLSTM

3.3.1. Convolution Neural Network

Convolution neural network [44] consists of an independent filter used for image data, classification prediction problems, and regression prediction problems due to its deep structure, it is also called DCNN. The number of features depends on the number of filters and extracts the mel-spectrogram of raw data (wav file). Each convolution layer learns features from the mel-spectrogram and the remaining layers process the useful information from the learned feature. This network consists of the 24-layer architecture of DCNN given in Figure 6. We have used the following Algorithm 1 for DCNN.

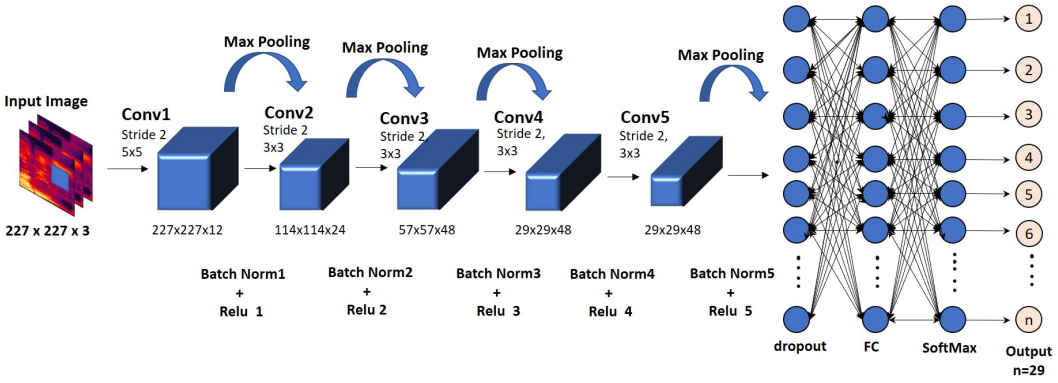


Figure 6. Deep convolution neural network architecture.

Algorithm 1: Classification task performed using DCNN.

Input
ads = Audio dataset
nLabels = Number of classes
Labels = Define class labels
numBands = Numbers of bands
Seg_Dr = Segment duration
Hop_Dr = Hop duration
Frame_Dr = Frame duration

Output
Accuracy = Model accuracy
YPredicted = Predicted labels

Algorithm
Begin
nBands, Seg_Dr, Hop_Dr, Frame_Dr←Define Parameters
adsTrain[ ], adsTest[ ] ← Split(ads)
melspectrogram(adsTrain[ ], Seg_Dr, Hop_Dr, Frame_D, nBands)→XTrain[ ]
melspectrogram (adsTest[ ], Seg_Dr, Hop_Dr, Frame_Dr, nBands)→ XTest[ ]
YTrain[ ] ← adsTest.Labels[ ]
YTest[ ] ← adsTest.Labels[ ]
Define Image Size → imageSize[ ]
trainNetwork ← XTrain[ ], YTrain[ ], layers, options
YPredicted[ ], Probability[ ] ← classify(trainedNetwork, XTest)
Accuracy ← mean (YPredicted[ ] == YTest[ ])
Confusion_Matrix ← YPredicted[ ], YTest[ ]
End

The transfer learning (TL) [45,46] technique is used in ML and its sub-field DL. This method is designed for one task and can be reused as a starting point for a new related task. Pre-trained networks are used as a starting point in new research, as these

networks help us save vast computation and time resources required to design a network. There are two ways to use transfer learning. Firstly, by extracting features using a pre-trained network, and then train the network model. Secondly, by fine-tuning the pre-trained network by keeping the weights learned as an initial parameter. Fine-tuning is used when we are using an NN that has been designed and trained by someone else. It allows taking advantage without having to develop it from scratch. Therefore, we are relying on the second method.

AlexNet is trained on millions of images from the ImageNet database [44]. It is trained in 1000 categories and is enriched with a wide range of feature representations. The standard size of this network is 227 × 227 × 3 (In image size 227 represents the number of frames, 227 represents the number of bands, and 3 represents the spectrum.) and consists of 25-layers shown in Figure 7. We convert the raw data to the mel-spectrogram because AlexNet is trained on the ImageNet dataset. The mel-spectrograms are resized according to AlexNet's standard input size, and inputs to the model. The standard AlexNet is trained on 1000 categories, whereas this work consists of 29 classes of the alphabet classification problem and 2 classes of the alphabet pronunciation classification problem. Therefore, to use pre-trained AlexNet, we have replaced 3 final layers of AlexNet named fully connected layer (Fc8), SoftMax layer, and classification layer (output layer). We have fine-tuned them according to our classification problems. After this, the network extracts features from the mel-spectrogram autonomously and learns from the dataset.

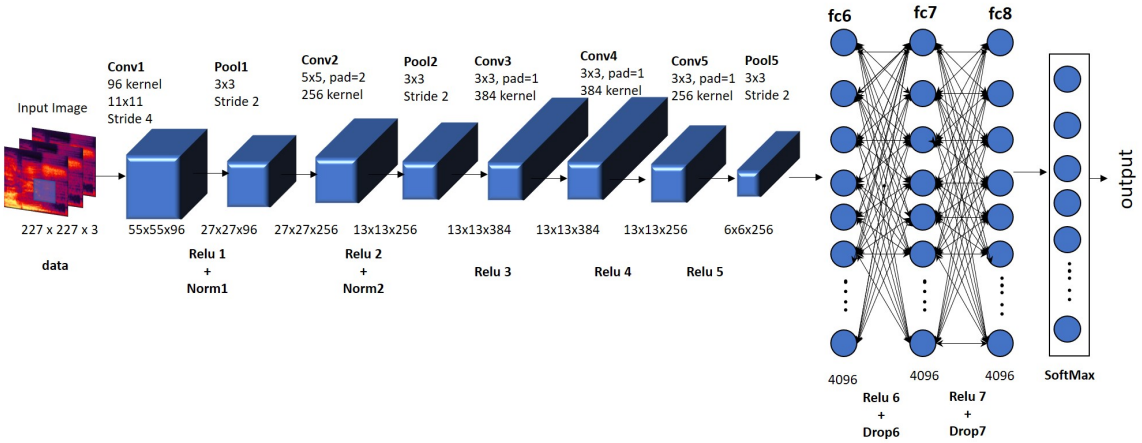


Figure 7. AlexNet architecture.

We need to specify the output size of the fully connected layers according to the number of classes of our data. Other parameters are learned after their empirical analysis, and network training. The test dataset is compared to the training dataset to observe the performance of the network. We have used Algorithm 2 for the AlexNet pre-trained network model.

3.3.2. Recurrent Neural Network

The recurrent neural network is designed to work with sequence prediction problems. Long short-term memory (LSTM) network is a special kind of RNN, which is skilled in learning long-term dependencies to help RNN in remembering long-term information lost during training.

Algorithm 2: Classification task using AlexNet with the transfer learning approach.

Input
ads = Audio dataset
nLabels = Number of classes
Labels = Define class labels
Output
AlexNet_Accuracy = Model accuracy
YPredicted = Predicted labels
Algorithm
Begin
XTrain[ ], YTrain[ ] ← Training set
XTest[ ], YTest[ ] ← Testing set
Define Image Size → inputSize[ ]
network → AlexNet
layersTranfer ← net.Layers(1:end-3)
netTransfer ← trainNetwork(XTrain, YTrain, layers_1, options)
[YPredicted, Probability] ← classify(netTransfer, XTest)
AlexNet_Accuracy ← mean(YPredicted[ ] == YTest[ ])
Confusion_Matrix ← confusionchart(YTest, YPred)
End

RNNs and LSTMs have received a high success rate when working with sequences of words and paragraphs. This includes both sequences of text and spoken words represented as time series. RNNs are mostly used for text data, speech data, classification prediction problems, regression prediction problems, and generative models. LSTM is a unidirectional network that learns only forward sequence as it can only see past. Whereas BLSTM is used to predict backward and forward sequence: one from past to future and the other from future to past [47]. We have used BLSTM for the training of our dataset and the architecture is shown in Figure 8. The most commonly used in audio sequences for spectral feature extraction for input in RNN networks. We have used the following Algorithm 3.

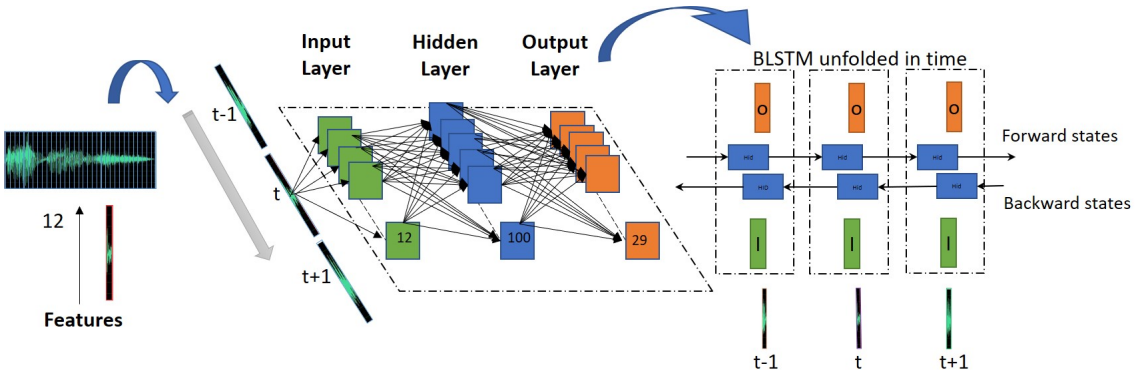


Figure 8. Bidirectional long short-term memory (BLSTM) architecture.

3.4. Classification of Unseen Data

Partitioning of the data is useful for the training and evaluation of machine learning models. The dataset is usually divided into two non-overlapping groups: training data and testing data. The training data are used for the modeling and feature set development. The test data are used to measure the model's performance. We have divided our dataset into 80% training data and 20% testing data.

Algorithm 3: Classification task using a recurrent neural network (BLSTM).

Input
ads = Audio dataset
nLabels = Number of classes
Labels = Define class labels
Output
YPredicted = Predicted labels
AlexNet_Accuracy = Model accuracy
Algorithm
Begin
FeatureTrain[ ] ← ExtractFeature(adsTrain), FeatureTest[ ] ← ExtractFeature(adsTest)
FeatureYTrain [ ] ← FeatureTrain.labels , FeatureYTest[ ] ← FeatureTest.Labels
Define Image Size → inputSize[ ]
for i ← 1:numObservationsTrain
sequence ← FeaturesTrain{i}
sequenceLengthsTrain(i) ← size(sequence, 2)
endfor
[sequenceLengthsTrain, idx] ← sort(sequenceLengthsTrain)
for i ← 1:numObservationsTest
sequence ← Featurestest{i}
sequenceLengthsTest(i) ← size(sequence, 2)
endfor
[sequenceLengthsTest, idx] ← sort(sequenceLengthsTest)
XTrain ← FeaturesTrain(idx), YTrain ← FeatureYTrain(idx)
XTest ← FeaturesTest(idx), YTest ← FeatureYTest(idx)
network → BLSTM
net ← trainNetwork(XTrain, YTrain, layers, options)
[YPredicted, Probability] ← classify(net, YTest)
BLSTM_Accuracy ← mean(YPredicted[ ] == YTest[ ])
Confusion_Matrix ← confusionchart(YTest, YPred)
End

4. Results and Discussion

We collected 8 audio samples from the web (see data availability section for dataset) and recorded 20 audio samples from native and non-native speakers at the sampling frequency of 44.1 kHz. We used a 16-bit pulse-coded modulated (PCM) raw format to collect audio samples. We collected 2 audio samples (per alphabet) from 11 male experts and 9 children (boys).

The dataset consists of speech samples from male subjects including 19 (adults) and 9 (children). The source data has 48 speech samples or 1392 files while augmented data has 3480 files. The description is given below in Table 2.

Table 2. Dataset of alphabet classification.

Resources	Speakers	Audio Samples	Age	Ethnicity
Collected from web	8 Male	1 (per speaker)	adult	unknown
Collected from expert	11 Male	2 (per speaker)	adult	5 natives, 6 non-natives
	9 Boys	2 (per speaker)	7 (10–14), 2 (15–18)	1 native, 8 non-natives
Source Data	48 samples	1392		
Augmented Data	120 samples (per alphabet)	3480		
Total dataset		4872		

A binary classification task is performed on correct versus non-correct pronunciation of the speech samples. We collected 20 samples from non-expert adults (male). Source data consist of 580 correct and 580 non-correct data samples. After augmentation, we have 140 samples per alphabet, and source data are 20 samples and augmented data of 120 samples per alphabet. This experiment needs an equal number of samples for correct versus non-correct pronunciation for the dataset so we selected source data of 20 samples and augmented data of 120 samples per alphabet of adult speakers. The description of the dataset is given in Table 3.

Table 3. Dataset for alphabet pronunciation classification.

Resources	**Speakers**	**Age**	**Ethnicity**
Collected from expert	20 Male (1 per speaker)	adults	non-native
Collected from novice	20 Male (1 per speaker)	adults	non-native
Dataset	**Source**	**Augmented**	**Total**
Correct pronunciation (experts)	20 samples	120 samples	4060 files
Mispronunciation (novice)	20 samples	120 samples	4060 files
Total Data			8120 files

4.1. Confusion Matrix

The confusion matrix is a performance measure of classification models [48], it consists of true positive (TP), true negative (TN), false positive (FP), and false negative (FN) measures. The accuracy (ACC) of the model in terms of aforementioned measures is:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \times 100 \quad (1)$$

Here, TP is the number of the alphabet that is positive (current class) and predicted correctly as positive, TN is the number of the alphabet that is negative (classes other than current class) and predicted correctly as negative, FP is the number of the negative alphabet that is negative but predicted incorrectly as positive, and FN is the number of the alphabet that is positive but predicted incorrectly as negative.

4.2. Validation Strategies

The validation step helps us find the optimum parameters for our model while preventing it from becoming overfitted. Two of the most known validation strategies are:

1. Hold-out strategy.
2. K-fold strategy.

In the hold-out validation [49], the dataset is divided into two non-overlapping sets of the training and testing dataset. The test dataset is held out while training the network model. It prevents overlapping and estimates the more accurate and generalized performance of an algorithm. It also reduces the computational cost because it only needs to be run once. The drawback of this procedure is that it does not use all the available data and results are highly dependent on the choice of training and testing data split.

Cross-validation [49] is a very powerful re-sampling technique used to evaluate ML models for limited data. It consists of a single parameter 'K' referred to as the number of groups in which the dataset is divided, this method is also referred to a K-fold cross-validation. We have performed 5 cross-fold validation, so it can use all the available data for validation.

4.3. Arabic Alphabet Classification

In this section, we are going to present the result of the dataset before data augmentation and after data augmentation for Arabic alphabet classification-based on Arabic alphabet pronunciation classification using the deep learning network models. This section

is concluded with a brief discussion of the comparison of the results of the deep learning models using DCNN, AlexNet, and BLTSM, respectively.

For the alphabet classification without data augmentation, the accuracy of the DCNN model is 65.89% with random splitting, 64.56% with the hold-out validation, and 64.03% with 5-fold CV. We trained the model 5 times by altering the neighboring sequence and then averaging out the accuracy. By using the TL approach, we have achieved an accuracy of 78.03% with random splitting, 78.73% with the hold-out validation, and we have achieved an accuracy of 79.15% with CV. BLSTM achieved 53.18% accuracy with random splitting, 52.62% with the hold-out validation, and after alternating sequences using 5-fold CV, the results are 53.17% as shown in Table 4.

For the alphabet classification with data augmentation, the accuracy of the DCNN model is 95.95% with random splitting, 93.32% with the hold-out validation, and 93.46% with 5-fold CV. The accuracy of the pre-trained AlexNet is 90.91% with SVM classifier which increased up to 98.41% using Adam optimizer with random splitting, 96.72% with the hold-out validation, and 96.36% with 5-fold CV. In BLSTM, we achieve an accuracy of 87.90% with random splitting, 88.38% with the hold-out validation, and 89.95% with the CV experiment as shown in Table 4. In Table 5 the values of mean, standard deviation (SD), and standard mean error (SME) are shown for DCNN, AlexNet, and BLSTM network.

Table 4. Alphabet classification with and without data augmentation.

Models	Without Data Augmentation			With Data Augmentation		
	Random Split	Hold-Out	5-Fold CV	Random Split	Hold-Out	5-Fold CV
DCNN	65.89%	64.56%	64.03%	95.95%	93.32%	93.46%
Alex Net(TL)	78.03%	78.73%	79.15%	98.41%	96.72%	96.36%
BLSTM	53.18%	52.62%	53.17%	87.90%	88.38%	87.95%

Table 5. Alphabet classification values of mean, standard deviation, and error.

Models	Mean	SD	Margin of Error
DCNN	94.63%	±1.17	±0.41
Alex Net(TL)	96.03%	±1.599	±0.57
BLSTM	87.025%	±1.95	±0.69

4.3.1. Without Data Augmentation

In this section, we are going to discuss the DCNN, AlexNet, and BLSTM network trained over the dataset without data augmentation in detail.

DCNN: The DCNN without data augmentation has an accuracy of 65.89%. The alphabet 'jeem', 'saud', and 'wao' have 100% accuracy, whereas 10 alphabets have accuracy above 70% and the other 10 alphabets have accuracy above 50% and the remaining alphabet have very low accuracy rate. The alphabet 'sa' is confused with 'hha', 'saud', 'fa', and 'ya', due to which these samples are misclassified.

Pre-trained AlexNet: The AlexNet network performed better than DCNN without data augmentation. The alphabet such as 'alif', 'jeem', 'dal', 'za', 'seen', 'sheen', 'duad', 'zua', 'aain', 'laam', 'meem', 'noon', 'wao', 'ya', and 'hamzah' have 100% accuracy. The 4 alphabets have accuracy above 80%, and the alphabet 'hha', 'tua', and 'ha' have accuracy less than 40%. The network is confusing 5 samples of 'ha' with 'sa' because the network is unable to classify these alphabets due to few data elements.

BLSTM: By using the BLSTM network, we have achieved an accuracy of 53.18% which is comparatively less than the other networks (DCNN and AlexNet). 'Alif' has 100% accuracy whereas only 2 alphabets have 80% accuracy and the other 10 alphabets have an accuracy between 70% and 60%. The remaining alphabet, i.e., 'sa' and 'fa', have the lowest accuracy of 16.7% because the network is confusing 'sa' with 'fa', 'ta', and 'ha'. The network is also confusing 'fa' with 'sa', 'za', and 'ha'.

4.3.2. With Data Augmentation

In this section, we are going to discuss the results of the DCNN, AlexNet, and BLSTM network trained over the whole dataset including source data and augmentation data in detail.

DCNN: The columns and rows represent the predicted class and actual class. The diagonal cells show the number of correctly classified observations and the off-diagonal cells show the number of incorrect observations. The alphabet 'alif', 'ba', 'zhal', 'za', 'seen', 'sheen', 'saud', 'aain', 'ghain', 'kaaf', 'meem', and 'hamzah' have 100% accuracy. Only the observations of the misclassified alphabet are shown in Table 6. The overall accuracy of the DCNN is 95.95%.

Table 6. DCNN with data augmentation.

seq.	sa	jeem	hha	kha	dal	zhal	za	duad	tua	zua	fa	qauf	kaaf	laam	meem	noon	wao	ya	hamzah	Acc
sa	28	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	96.3%
jeem	0	31	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	96.7%
hha	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	93.8%
kha	0	0	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	96.6%
dal	0	0	0	0	31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	96.7%
zhal	1	0	0	0	0	31	0	0	0	0	0	0	1	0	0	0	0	0	0	96.7%
za	0	0	0	0	0	0	28	0	0	2	0	1	0	0	0	0	0	0	1	86.7%
duad	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	96.8%
tua	0	0	0	0	0	0	0	1	30	0	0	0	0	0	0	0	0	0	0	90%
zua	0	0	0	0	0	0	1	0	1	29	0	0	0	0	0	0	0	0	1	90%
qauf	0	1	0	0	0	0	0	0	0	0	31	0	0	0	0	0	0	0	0	96.6%
kaaf	0	0	0	0	0	0	1	0	0	1	0	29	0	0	0	0	0	0	0	96.4%
laam	0	0	0	0	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	93.8%
meem	0	0	0	0	0	1	0	0	0	0	0	0	0	31	0	0	0	0	0	93.5%
noon	0	0	2	0	0	0	0	0	0	0	0	0	0	0	30	0	0	0	0	100%
ha	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30	0	2	0	93.3%
wao	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	27	1	0	96.4%
ya	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	29	0	87.1%
hamzah	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	30	93.5%

In cross validation, we have applied a 5-fold validation experiment on the DCNN model. We cross-check and validate the non-overlapping data k-fold times. The alphabet 'alif', 'zua', 'qauf', 'wao', and 'hamzah' have 100% accuracy. The accuracy of 93.46% by splitting the train and test datasets into 80% and 20%.

Pre-trained AlexNet: By using pre-trained AlexNet model, the results achieved are better than the DCNN network. We achieved an accuracy of 98.41% with random split. We can see from Table 7 that the network is confusing two sample 'fa' with 'sa'. One sample of 'duad' with 'zua', 'za' with 'zua', 'tua' with 'zua', 'qauf' with 'kaaf', 'ha' with 'hha' and 'meem' with 'jeem' and vice versa. Accurately classified classes are 'alif', 'ba', 'sa', 'jeem', 'ra', 'seen', 'sheen', 'suad', 'aain', 'ghain', 'laam', 'noon', 'wao', 'ya', 'hamzah' having 100% accuracy. Whereas the network provides an accuracy of 96.72% using hold-out validation.

Table 7. AlexNet with data augmentation.

seq.	sa	jeem	hha	kha	dal	zhal	ra	seen	sheen	tua	zua	aain	qauf	kaaf	laam	meem	ya	Acc
sa	31	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	96.4%
jeem	0	33	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	100%
hha	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	1	0	100%
kha	0	0	0	29	1	0	0	0	0	0	0	0	0	0	0	0	2	94.40%
dal	0	0	0	1	32	0	0	0	0	0	1	0	0	0	0	0	0	96.7%
zhal	0	0	0	0	0	32	1	0	0	0	0	0	0	0	0	0	0	98.6%
ra	0	0	0	0	0	0	31	0	0	1	0	0	0	0	0	0	0	96.7%
seen	0	0	0	0	0	0	0	30	1	0	0	1	0	0	0	0	0	96.7%
sheen	0	0	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	100%
tua	0	0	0	0	0	0	0	0	0	30	1	1	0	0	0	0	0	98.6%
zua	1	0	0	0	0	0	0	0	0	0	31	1	0	0	0	0	0	96.4%
aain	0	0	0	0	0	0	1	0	0	0	1	32	0	0	0	0	0	94.8%
qauf	0	0	0	0	0	0	0	1	0	0	0	0	30	0	0	0	0	96.4%
kaaf	0	0	0	0	0	0	0	0	0	0	0	0	0	31	0	0	0	96.6%
laam	0	0	0	0	0	0	0	0	0	0	0	0	0	2	31	0	0	98.4%
noon	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	31	0	98.4%
ya	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	29	94.8%

By using 5-fold CV experiment using AlexNet, the model accurately classified the alphabet 'alif', 'ba', 'jeem', 'za', 'seen', 'sheen', 'suad', 'zua', 'qauf', 'tua', 'noon', 'wao', 'ya' and 'hamzah' have 100% accuracy. Overall accuracy of the network with 5-fold CV is 96.36%, respectively.

BLSTM: In the BLSTM network, we achieved an accuracy of 87.90% with a random split and 88.38% with the hold-out validation. BLSTM is confusing some alphabet of matching sounds, i.e., 'ba', 'ta', 'sa', 'dal', 'zhal', 'tua', 'zua', 'ghain', 'fa', etc. The alphabet 'kha' and 'seen' are accurately classified with 100% accuracy.

By using a 5-fold CV experiment using the BLSTM network, we have achieved an accuracy of 87.95%. 'Kha' is the only alphabet that is accurately classified with 100% accuracy.

4.4. Arabic Alphabet Pronunciation Classification

For Arabic alphabet pronunciation classification without data augmentation, the accuracy of the DCNN model is 96.41% with random split, 95.46% with the hold-out, and 96.37 with the 5-fold CV experiment. The AlexNet transfer learning approach achieved an accuracy of 97.12%, 96.89% with the hold-out, and 96.55% with CV. The BLSTM achieved 72.41% accuracy with random split, 73.54% with the hold-out, and 74.35% with 5-fold CV as shown in Table 8.

For Arabic alphabet pronunciation classification with data augmentation using the DCNN model, we achieved an accuracy of 97.88% with the random split method, 95.28% with the hold-out validation, and 96.24% with 5-fold CV. The AlexNet transfer learning model achieved an accuracy of 99.14% with random split, 97.42% with the hold-out validation, and 98.43% with CV. The BLSTM model achieved an accuracy of 77.71% with random split, 76.12% with the hold-out validation, and 78.17% with 5-fold CV. In Table 9, the values of mean, standard deviation (SD), and standard mean error (SME) are shown for DCNN, AlexNet, and BLSTM.

Table 8. Arabic alphabet pronunciation classification with and without data augmentation.

Model	Without Data Augmentation			With Data Augmentation		
	Random Split	Hold-Out	5-Fold CV	Random Split	Hold-Out	5-Fold CV
DCNN	96.41%	95.46%	96.37%	97.88%	95.28%	96.24%
Alex Net(TL)	97.12%	96.89%	96.55%	99.14%	97.42%	98.43%
BLSTM	72.41%	73.54%	74.35%	77.71%	76.12%	78.17%

Table 9. Alphabet classification values of mean, standard deviation, and error.

Models	Mean	SD	Margin of Error
DCNN	95.03%	±2.57	±0.90
Alex Net(TL)	97.36%	±1.64	±0.58
BLSTM	73.21%	±2.40	±0.80

4.4.1. Without Data Augmentation

The confusion matrix of the Arabic alphabet pronunciation classification shows the number of correctly classified and misclassified samples. The accuracy of these samples can be seen in Table 10, it shows the confusion matrix of DCNN, AlexNet, and BLSTM models for the dataset without data augmentation. AlexNet has a lesser error rate than DCNN and BLSTM.

Table 10. DCNN without data augmentation.

Model		Predicted		
		Class	**Correct Pronunciation**	**Mispronunciation**
DCNN	Actual	Correct pronunciation	113	3
		Mispronunciation	3	113
AlexNet		Correct pronunciation	116	0
		Mispronunciation	3	113
BLSTM		Correct pronunciation	99	17
		Mispronunciation	47	69

4.4.2. With Data Augmentation

The dataset is split into a ratio of 80% training set and 20% test data. This approach consists of two classes because of less interference between classes. As shown in Table 11 both classes in DCNN have an accuracy of more than 97%, confusing 11 samples of Correct pronunciation class and 24 of mispronunciation class. Whereas, AlexNet has a few misclassified samples as compared to DCNN and BLSTM networks. The AlexNet provides the highest accuracy in pronunciation classification.

Table 11. DCNN without data augmentation.

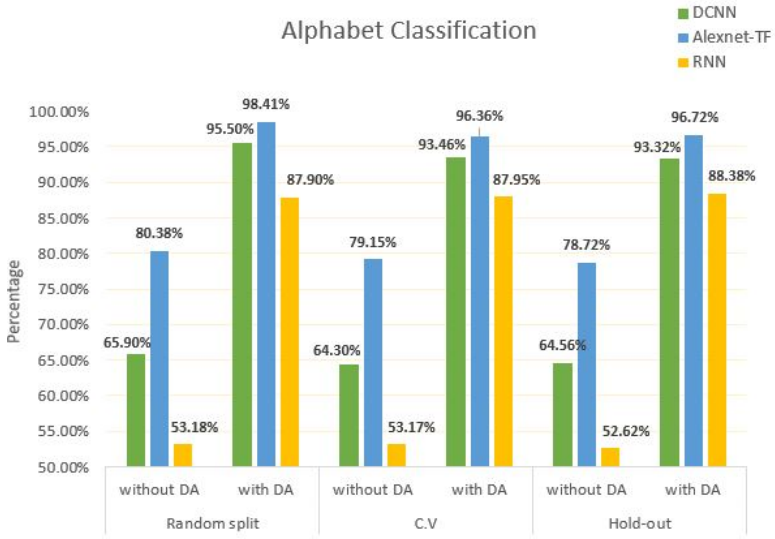
Model		Predicted		
		Class	**Correct Pronunciation**	**Mispronunciation**
DCNN	Actual	Correct pronunciation	788	24
		Mispronunciation	11	801
AlexNet		Correct pronunciation	806	6
		Mispronunciation	8	804
BLSTM		Correct pronunciation	781	31
		Mispronunciation	331	481

4.5. Discussion

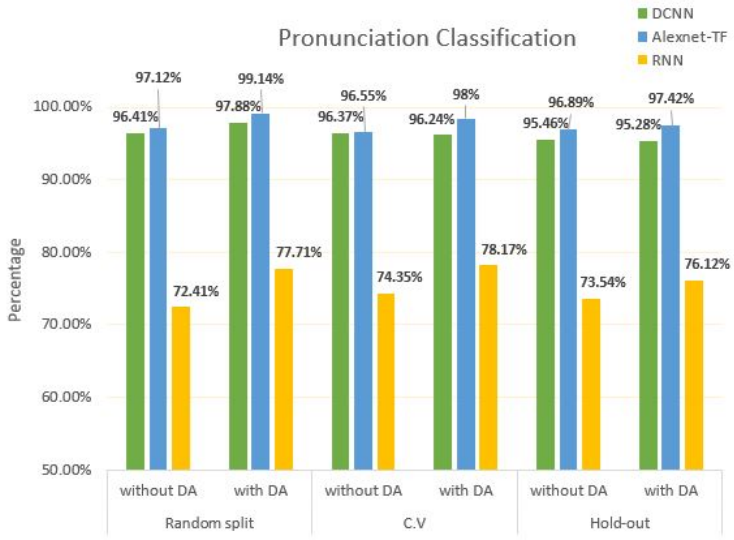
The classification model's performance with data augmentation outperformed the model's performance without data augmentation. We can increase the performance of the network by reducing overfitting and improving the accuracy of the network as can be seen in Figure 9. In this figure, the left bar represents DCNN, the middle one represents AlexNet, and the right one represents the BLSTM network tested on two validation techniques including 5-fold CV, and the hold-out validation for the dataset with data augmentation and without data augmentation. Figure 9a shows alphabet classification results with and without data augmentation and Figure 9b shows pronunciation classification results with and without data augmentation. The results in Figure 9b are so close because it has only two classes, by increasing the number of classes we might see the difference very clearly.

We can also see from the results given in previous sections that the AlexNet with transfer learning outperformed other networks, although its architecture is similar but deeper than the DCNN network. As a result that AlexNet is trained with 60 million parameters, the spectrograms are augmented by mirroring and cropping the images which increases variation in the training dataset. It uses overlapped pooling layers after some convolution layers which improves the error rate as compared to other networks. DCNN is the second network in the lead, we construct a small DCNN network as an array of layers seen in Figure 6 and then trained it from scratch. Due to over-fitting while training, its error rate is more than the AlexNet network. Now, the third network (BLSTM) has accuracy less than the other networks because it cannot extract features on its own while

both DCNN and AlexNet network extracts numerous features on their own from the spectrograms. Whereas, while using BLSTM we first extract the fixed number of features and then train it by using BLSTM, which is why its results are less satisfying than the other networks. The comparison of the ANNs with and without data augmentation for alphabet classification and alphabet pronunciation classification can be seen in Figure 9.



(a)



(b)

Figure 9. Comparison of DCNN, AlexNet, and BLSTM models with and without data augmentation using different validation strategies. (**a**) Alphabet classification. (**b**) Pronunciation classification.

5. Conclusions

In this paper, we have proposed a framework for CA speech recognition using deep learning techniques including DCNN, AlexNet, and BLSTM. We implemented these learning models and demonstrated their results on the Arabic alphabet audio dataset. Several experiments are performed using three different validation techniques including random splitting, 5-fold cross-validation, and hold-out validation. AlexNet outperformed the DCNN and BLSTM in the classification tasks. We have performed two tasks, i.e., Arabic alphabet classification and Arabic alphabet pronunciation classification using augmented and non-augmented dataset while we have achieved promising results with data augmentation.

The first part of this research is Arabic alphabet classification, which is successfully performed by using AlexNet and yielded an accuracy of 98.41% with data augmentation.

The second part of this research is the Arabic alphabet pronunciation classification using the AlexNet model and, we achieved an accuracy of 99.14% with data augmentation. As future work, we would like to extend the proposed method to incorporate more feature sets and increase the size of the dataset for words and sentence recognition. We would further like to investigate some new network architectures, i.e., Xception, Inception, ResNet, and NASNet.

Author Contributions: Conceptualization, H.F.A. and K.R.; methodology, N.Z., H.F.A., K.R., I.F., M.Z. and A.A.; software, N.Z.; validation, N.Z.; formal analysis, N.Z., K.R., and I.F.; resources, H.F.A.; data curation, N.Z. and H.F.A.; writing—original draft preparation, N.Z.; writing—review and editing, H.F.A., K.R., I.F., M.Z. and A.A.; visualization, N.Z. and K.R. All authors have read and agreed to the published version of the manuscript.

Funding: Deanship of Scientific Research, King Faisal University, Saudi Arabia.

Institutional Review Board Statement: The study has been performed as per the ethical standards as laid down in the 1964 Declaration of Helsinki and its later amendments or comparable ethical standards and approved by Institutional Review Board (or Ethics Committee) of Qauid-I-Azam University Islamabad, Pakistan (7 July 2020).

Informed Consent Statement: Informed consent was obtained from the guardians of minor participants and all the individual participants included in this research work while collecting the audio dataset.

Data Availability Statement: The wav data format is used to support the findings of this study. The dataset is collected from very few online resources: 1. http://www.Arabicquick.com/learn_Arabic_alphabet/ (accessed on 4 April 2019); 2. https://languagehub.co.nz/reading-writing-arabic-alphabet-audio-downloads/ (accessed on 28 March 2019); 3. https://www.searchtruth.com/quran/teacher/1/ (accessed on 4 April 2019). Due to the limited availability of the audio data regarding our research, we have collected our own dataset with the help of Hafiz Farooq Ahmad and Muhammad Zia. Dataset available at: https://drive.google.com/drive/folders/1GXyFO6LGBgO-FNRjDIu5fRALkoAb2en2 (accessed on 4 April 2019).

Acknowledgments: This work is supported by the Deanship of Scientific Research, King Faisal University, Al-Ahsa, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Julian, G. The 10 Most Spoken Languages In The World, May 2018. Available online: https://www.fluentin3months.com/most-spoken-languages/ (accessed on 20 November 2019).
2. Kher, J. The History of Arabic Language, November 2018. Available online: https://www.verbling.com/articles/post/the-history-of-arabic-language?locale=en (accessed on 22 November 2019).
3. Aqel, M.J.; Zaitoun, N.M. Tajweed: An Expert System for Holy Qur'an Recitation Proficiency. *Procedia Comput. Sci.* **2015**, *65*, 807–812. [CrossRef]
4. Wang, D.; Wang, X.; Lv, S. An Overview of End-to-End Automatic Speech Recognition. *Symmetry* **2019**, *11*, 1018. [CrossRef]
5. McGraw, I.; Prabhavalkar, R.; Alvarez, R.; Arenas, M.G.; Rao, K.; Rybach, D.; Alsharif, O.; Sak, H.; Gruenstein, A.; Beaufays, F.; et al. Personalized speech recognition on mobile devices. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP, Shanghai, China, 20–25 March 2016; pp. 5955–5959. [CrossRef]
6. Hewett, T.T.; Baecker, R.; Card, S.; Carey, T.; Gasen, J.; Mantei, M.; Perlman, G.; Strong, G.; Verplank, W. *ACM SIGCHI Curricula for Human-Computer Interaction*; Association for Computing Machinery: New York, NY, USA, 1992; p. 173. [CrossRef]
7. Clark, L.; Doyle, P.; Garaialde, D.; Gilmartin, E.; Schlögl, S.; Edlund, J.; Aylett, M.; Cabral, J.; Munteanu, C.; Edwards, J.; et al. The State of Speech in HCI: Trends, Themes and Challenges. *Interact. Comput.* **2019**, *31*, 349–371. [CrossRef]
8. Allan, J. Perspectives on information retrieval and speech. In Proceedings of the Workshop on Information Retrieval Techniques for Speech Applications, New Orleans, LA, USA, 13 September 2001; pp. 1–10. [CrossRef]
9. Stockwell, G. *Computer-Assisted Language Learning: Diversity in Research and Practice*; Cambridge University Press: Cambridge, UK, 2012; p. 213. [CrossRef]
10. Chien, J.T.; Chueh, C.H. Joint acoustic and language modeling for speech recognition. *Speech Commun.* **2010**, *52*, 223–235. [CrossRef]
11. Nassif, A.B.; Shahin, I.; Attili, I.; Azzeh, M.; Shaalan, K. Speech recognition using deep neural networks: A systematic review. *IEEE Access* **2019**, *7*, 19143–19165. [CrossRef]

12. Proença, J.; Lopes, C.; Tjalve, M.; Stolcke, A.; Candeias, S.; Perdigao, F. Mispronunciation Detection in Children's Reading of Sentences. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2018**, *26*, 1207–1219. [CrossRef]
13. Li, K.; Qian, X.; Meng, H. Mispronunciation Detection and Diagnosis in L2 English Speech Using Multidistribution Deep Neural Networks. *IEEE/ACM Trans. Audio Speech Lang. Process.* **2017**, *25*, 193–207. [CrossRef]
14. Wang, W.Y.; Wang, L.; Li, C.; Huang, Y. Using Extended Letter-to-Sound Rules to Detect Pronunciation Errors Made by Chinese Learner of English. In Proceedings of the 2010 International Conference on Computational Intelligence and Software Engineering, Wuhan, China, 10–12 December 2010; pp. 1–4.
15. Chen, B.; Hsu, Y.C., Mandarin Chinese Mispronunciation Detection and Diagnosis Leveraging Deep Neural Network Based Acoustic Modeling and Training Techniques. In *Computational and Corpus Approaches to Chinese Language Learning*; Lu, X.; Chen, B., Eds.; Springer: Singapore, 2019; pp. 217–234. [CrossRef]
16. Zhang, F.; Huang, C.; Soong, F.K.; Chu, M.; Wang, R. Automatic mispronunciation detection for Mandarin. In Proceedings of the 2008 IEEE International Conference on Acoustics, Speech and Signal Processing, Las Vegas, NV, USA, 30 March–4 April 2008; pp. 5077–5080.
17. Jurafsky, D.; Martin, J. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*; Prentice Hall: Upper Saddle River, NJ, USA, 2008; Volume 2.
18. Nadeu, C.; Macho, D.; Hernando, J. Time and frequency filtering of filter-bank energies for robust HMM speech recognition. *Speech Commun.* **2001**, *34*, 93–114. [CrossRef]
19. Reynolds, D. Gaussian mixture models. In *Encyclopedia of Biometrics*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 659–663. [CrossRef]
20. Da Silva, I.N.; Spatti, D.H.; Flauzino, R.A.; Liboni, L.H.B.; dos Reis Alves, S.F. *Artificial Neural Networks*; Springer International Publishing: Cham, Switzerland, 2017. [CrossRef]
21. Yang, S.; Cao, J.; Wang, J. Acoustics recognition of construction equipments based on LPCC features and SVM. In Proceedings of the 2015 34th IEEE Chinese Control Conference (CCC), Hangzhou, China, 28–30 July 2015, pp. 3987–3991. [CrossRef]
22. Walker, W.; Lamere, P.; Kwok, P.; Raj, B.; Singh, R.; Gouvea, E.; Wolf, P.; Woelfel, J. Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Available online: https://www.researchgate.net/publication/228770826_Sphinx-4_A_flexible_open_source_framework_for_speech_recognition (accessed on 4 April 2019).
23. Abushariah, M.A.; Ainon, R.N.; Zainuddin, R.; Elshafei, M.; Khalifa, O.O. Natural speaker-independent Arabic speech recognition system based on hidden Markov models using Sphinx tools. In Proceedings of the International Conference on Computer and Communication Engineering (ICCCE'10), Kuala Lumpur, Malaysia, 11–13 May 2010; pp. 1–6. [CrossRef]
24. Abushariah, M.A.A.M.; Ainon, R.N.; Zainuddin, R.; Elshafei, M.; Khalifa, O.O. Arabic speaker-independent continuous automatic speech recognition based on a phonetically rich and balanced speech corpus. *Int. Arab. J. Inf. Technol. (IAJIT)* **2012**, *9*, 84–93.
25. Satori, H.; Harti, M.; Chenfour, N. Arabic speech recognition system based on CMU Sphinx. In Proceedings of the 2007 International Symposium on Computational Intelligence and Intelligent Informatics, Agadir, Morocco, 28–30 March 2007; pp. 31–35.
26. Muhammad, A.; ul Qayyum, Z.; Tanveer, S.; Martinez-Enriquez, A.; Syed, A.Z. E-hafiz: Intelligent system to help Muslims in recitation and memorization of Quran. *Life Sci. J.* **2012**, *9*, 534–541.
27. Arshad, N.W.; Sukri, S.M.; Muhammad, L.N.; Ahmad, H.; Hamid, R.; Naim, F.; Naharuddin, N.Z.A. Makhraj recognition for Al-Quran recitation using MFCC. *Int. J. Intell. Inf. Process. (IJIIP)* **2013**, *4*, 45–53.
28. Ahsiah, I.; Noor, N.; Idris, M. Tajweed checking system to support recitation. In Proceedings of the International Conference on Advanced Computer Science and Information Systems (ICACSIS), Sanur Bali, Indonesia, 28–29 September 2013; pp. 189–193. [CrossRef]
29. Altalmas, T.; Ahmad, S.; Sediono, W.; Hassan, S.S. Quranic letter pronunciation analysis based on spectrogram technique: A case study on Qalqalah letters. In Proceedings of the 11th International Conference on Artificial Intelligence Applications and Innovations (AIAI'15), Bayonne, France, 14–17 September 2015; Volume 1539, pp. 14–22.
30. Maqsood, M.; Habib, H.A.; Nawaz, T.; Haider, K.Z. A complete mispronunciation detection system for Arabic phonemes using SVM. *Int. J. Comput. Sci. Netw. Secur. (IJCSNS)* **2016**, *16*, 30.
31. Lauzon, F.Q. An introduction to deep learning. In Proceedings of the 11th International Conference on Information Science, Signal Processing and their Applications (ISSPA), Montreal, QC, Canada, 2–5 July 2012; pp. 1438–1439. [CrossRef]
32. Ahmad, A.M.; Ismail, S.; Samaon, D. Recurrent neural network with backpropagation through time for speech recognition. In Proceedings of the IEEE International Symposium on Communications and Information Technology (ISCIT), Sapporo, Japan, 26–29 October 2004; Volume 1; pp. 98–102. [CrossRef]
33. Nazir, F.; Majeed, M.N.; Ghazanfar, M.A.; Maqsood, M. Mispronunciation Detection Using Deep Convolutional Neural Network Features and Transfer Learning-Based Model for Arabic Phonemes. *IEEE Access* **2019**, *7*, 52589–52608. [CrossRef]
34. Lederle, M.; Wilhelm, B. Combining High-Level Features of Raw Audio Waves and Mel-Spectrograms for Audio Tagging. *arXiv* **2018**, arXiv:1811.10708.
35. Roh, Y.; Heo, G.; Whang, S.E. A survey on data collection for machine learning: A big data-ai integration perspective. *IEEE Trans. Knowl. Data Eng.* **2019**, *33*, 1328–1347. [CrossRef]
36. Asadullah, M.; Nisar, S. A silence removal and endpoint detection approach for speech processing. *Sarhad Univ. Int. J. Basic Appl. Sci.* **2017**, *4*, 10–15. [CrossRef]

37. Singh, C.; Venter, M.; Muthu, R.K.; Brown, D. Chapter 3—A Real-Time DSP-Based System for Voice Activity Detection and Background Noise Reduction. In *Intelligent Speech Signal Processing*; Dey, N., Ed.; Academic Press: Cambridge, MA, USA, 2019; pp. 39–54. [CrossRef]
38. Giannakopoulos, T. A Method for Silence Removal and Segmentation of Speech Signals, Implemented in Matlab. Ph.D. Thesis, University of Athens, Athens, Greece, 2009; Volume 2.
39. Mikołajczyk, A.; Grochowski, M. Data augmentation for improving deep learning in image classification problem. In Proceedings of the International Interdisciplinary PhD workshop (IIPhDW), Swinoujscie, Poland, 9–12 May 2018; pp. 117–122. [CrossRef]
40. Elnemr, H.A.; Zayed, N.M.; Fakhreldein, M.A. Feature extraction techniques: fundamental concepts and survey. In *Handbook of Research on Emerging Perspectives in Intelligent Pattern Recognition, Analysis, and Image Processing*; IGI Global: Hershey, PA, USA, 2016; pp. 264–294.
41. Su, Y.; Zhang, K.; Wang, J.; Madani, K. Environment sound classification using a two-stream CNN based on decision-level fusion. *Sensors* **2019**, *19*, 1733. [CrossRef]
42. Sherstinsky, A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *arXiv* **2018**, arXiv:1808.03314.
43. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
44. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in nEural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2012; Volume 1, pp. 1097–1105. [CrossRef]
45. López-Sánchez, D.; Arrieta, A.G.; Corchado, J.M. Deep neural networks and transfer learning applied to multimedia web mining. In Proceedings of the 14th International Symposium on Distributed Computing and Artificial Intelligence (DCAI), Porto, Portugal, 21–23 June 2017; pp. 124–131. [CrossRef]
46. Aghamaleki, J.A.; Baharlou, S.M. Transfer learning approach for classification and noise reduction on noisy web data. *Expert Syst. Appl.* **2018**, *105*, 221–232. [CrossRef]
47. Qayyum, A.; Latif, S.; Qadir, J. Quran reciter identification: A deep learning approach. In Proceedings of the 7th International Conference on Computer and Communication Engineering (ICCCE), Kuala Lumpur, Malaysia, 19–20 September 2018; pp. 492–497.
48. Silva-Palacios, D.; Ferri, C.; Ramírez-Quintana, M.J. Improving performance of multiclass classification by inducing class hierarchies. *Procedia Comput. Sci.* **2017**, *108*, 1692–1701. [CrossRef]
49. Refaeilzadeh, P.; Tang, L.; Liu, H. Cross-validation. In *Encyclopedia of Database Systems*; Springer: New York, NY, USA, 2009; pp. 532–538.