Received: 19 July 2021 | Revised: 31 August 2021 | Accepted: 4 September 2021

DOI: 10.1049/tje2.12082

ORIGINAL RESEARCH PAPER

How many Mel-frequency cepstral coefficients to be utilized in speech recognition? A study with the Bengali language

Md. Rakibul Hasan[1,2] | **Md. Mahbub Hasan**[1] | **Md Zakir Hossain**[3,4]

[1] Department of Electrical and Electronic Engineering, Khulna University of Engineering & Technology, Khulna, Bangladesh

[2] Department of Electrical and Electronic Engineering, BRAC University, Dhaka, Bangladesh

[3] Agriculture and Food, ML&AI FSP, Commonwealth Scientific and Industrial Research Organisation, Black Mountain, Canberra, Australia

[4] Biological Data Science Institute, Australian National University, Canberra, Australia

Correspondence
Md. Mahbub Hasan, Department of Electrical and Electronic Engineering, Khulna University of Engineering & Technology, Khulna 9203, Bangladesh.
Email: mahbub01@eee.kuet.ac.bd

Abstract
Speech-related research has a wide range of applications. Most speech-related researches employ Mel-frequency cepstral coefficients (MFCCs) as acoustic features. However, finding the optimum number of MFCCs is an active research question. MFCC-based speech classification was performed for both vowels and words in the Bengali language. As for the classification model, deep neural network (DNN) with Adam optimizer was used. The performances were measured with five different performance metrics, namely confusion matrix, classification accuracy, area under curve of receiver operating characteristic (AUC-ROC), F_1 score, and Cohen's Kappa with four-fold cross-validations at different number of MFCCs. All performance metrics gave the best score for 24/25 MFCCs; hence it is suggested that the optimum number of MFCCs should be 25, although many existing studies use only 13 MFCCs. Furthermore, it is verified that increasing the number of MFCCs yields better classification metrics with lower computational burden than the increment of hidden layers. Lastly, the optimum number of MFCCs obtained from this study was used in a more improved DNN model, from which 99% and 90% accuracies were achieved for vowel and word classification, respectively, and the vowel classification score outperformed state-of-the-art results.

1 | INTRODUCTION

Speech processing-based systems such as Microsoft Cortana, Google Assistant, and Amazon Alexa have vastly simplified our modern life. These command-based services are a helping hand not only for ordinary people but also for physically challenged and old-age people. In general, speech recognition devices and systems have made several automations in our lives—home automation, automation in smartphones, laptops, and vehicles. Apart from these devices and systems, speech processing is making a significant impact in healthcare, such as diagnosis of Parkinson's disease [1], Dementia [2], and Alzheimer's disease [3]. Other applications include wearable for deaf persons [4], keyword spotting [5], emotion recognition [6], accent classification [7], and language identification [8]. Accordingly, research employing speech processing is booming with a particular interest in speech recognition, classification, and generation.

The authors of [9] specify some standard features used in phone-based speech recognition systems. In general, the standard spectral features, including linear prediction cepstral coefficient (LPCC) and Mel-frequency cepstral coefficient (MFCC) representing the gross shape of the vocal tract, are the most widely used. For example, a recent survey paper [10] reports 24 contemporary works on the Bengali language, out of which 15 works have used MFCC as input features. Another survey paper on Arabic speech recognition [11] reports 13 studies on isolated words speech recognition, out of which 12 studies have used MFCC. Furthermore, da Silva et al. [12] state that the best recognition performance can be achieved by utilizing MFCC. Therefore, MFCC is the feature of significant interest in most speech-related researches.

The broad demand for MFCC can be proved from several studies in the Bengali language. The authors of [13] classified ten spoken Bengali digits by using MFCC features. In a Bengali speech corpus development, Das et al. [14] performed phoneme recognition using 13 base MFCCs and their first and second-order derivatives. The authors of [15] also used the same features to develop a Sphinx3-based Bengali speech recognition system with the main focus to help visually impaired people. The authors of [16] performed isolated Bengali speech recognition employing MFCC features. Furthermore, Sumon et al. [17] experimented classification of ten Bengali short speech words based on MFCC features and raw audio files in two different classification models. They found that MFCC-based classification outperforms raw audio-based classification. The authors of [18] performed recognition of Bengali speech characters (vowels and consonants) using MFCC features. On top of these, Badhon et al. [19] remarked some state-of-the-art studies in Bengali Speech Recognition up to the year of 2019. Their study reveals that MFCC and linear prediction coefficients are the most adopted feature extraction techniques.

Apart from the Bengali language, MFCC has been used in other language researches as well. Most of them utilized 13 MFCC features in a deep neural network (DNN) classifier, such as recognition of speakers [20], English phoneme [21], emotion from English audio [6], five Malayalam vowel phonemes [22], twenty Dari speech tokens [23], three Arabic words [24], and ten English command words [25]. Some studies also prefer convolutional neural network (CNN) models such as isolated English word recognition by Soliman et al. [26]. The authors of [27] developed a real-time speech emotion recognition system from continuous speech utilizing three cepstral features—perceptual linear prediction (PLP) cepstral coefficients, MFCC, and LPCC. Furthermore, Salau et al. [7] performed accent classification of three Nigerian languages using MFCC features. From their comparison with similar studies, it is evident that MFCC is the most accepted feature.

Selection of the number of coefficients in any appropriate feature is as essential as the selection of classifiers, and in fact, classification performance employing MFCC is undoubtedly dependent on the optimum number of coefficients [28]. Our main contribution is to find out the optimum number of MFCCs to extract the best possible score. Classification performance can be improved by incrementing the number of MFCCs and the number of hidden layers. More MFCC features indicate more acoustic information from speech. Therefore, when more MFCCs are used, DNN's computational burden increases for classifying information associated with additional MFCCs. Similarly, increasing the number of hidden layers also raises the computational burden of DNN. Therefore, a question arises: which one (MFCC increase vs. hidden layer increase) can produce a better result with a minimum computational burden? To the best of the authors' knowledge, very few articles [12, 28] address similar questions and related issues. We have reported on this issue based on the Bengali language. Beside this, we developed a more improved DNN-based classification model to pull out the best possible classification score utilizing the optimum number of MFCC. Comparison with other relevant studies reveals the competitiveness of our classification scores. Since vowels and consonants are the fundamental building blocks of any language, proper detection of vowels and consonants holds primary importance for any speech recognition system of that language. Moreover, a combination of vowels and consonants produces speech sounds that hold necessary information to communicate. Thus, the outcome of this research will help increase the performance of MFCC dependent systems.

This article is organized as follows. First, Section 2 presents the datasets, MFCC extraction process, DNN-based classification model, and overall workflow. Then, Section 3 can be broadly classified into four subsections. Firstly, we tried to find the optimum number of MFCC. Secondly, we compared the suitability between MFCC increase and hidden layer increase. Thirdly, we varied the DNN configuration to find out the best possible score with the optimum number of MFCCs, and finally, we compared both the performance score and the number of MFCCs used in related studies. Furthermore, all findings and applications are summarized in Section 4.

2 | MATERIAL AND METHODS

This section describes the datasets and core methodologies involved. A sketch of the steps involved is depicted in Figure 1.

To perform the three types of experiments mentioned in the Introduction section (finding the optimum number of MFCC; suitability between MFCC and hidden layer increase; and securing the best possible score), we have first designed the experimental setups. For the classification of vowels and words, DNN architecture is utilized, and the structure is also tuned to obtain better classification results. Tuning involves the optimum number of hidden layers, number of neurons in them, activation functions of respective layers, loss function, optimizer and its learning rate, batch mode training, batch size, number of epochs, train-test split ratio, and evaluation metrics or accuracy indexes. In all these cases, we had multiple options, from where we have selected optimum one by tracking the evaluation metrics. The descriptions of these steps are given in the following subsections.

2.1 | Speech collection and dataset preparation

We have two datasets—one for vowel classification and another for word classification [29]. Seven Bengali vowels (/অ/[/ɔ/], /আ/[/a/], /ই/[/i/], /উ/[/u/], /ঋ/[/ri/], /এ/[/e/], and /ঐ/[/oi/]) and seven Bengali words (বোতল, বন, কপি, দোকান, শেষ, সঠিক, and উপরে) were selected for vowel classification and word classification, respectively. We collected these data from 20 Bangladeshi speakers (age 20–26 years), and Bengali is their first language. Guiding the speakers to pronounce the speeches in two different accents, we have collected 40 sound signals corresponding to each of these seven vowels and words. We recorded the sounds on a "Xiaomi Redmi 3" smartphone and later processed them in version 2.2.2 of the Audacity software [30]. Accordingly, we have created the two datasets, where we have 40 utterances in each of the seven classes for both vowels and words. Therefore, each dataset consists of $40 \times 7 = 280$ samples, and considering two datasets (vowels and words), we have 560 samples in total. Figure 2 presents the waveshape of two sample sounds from

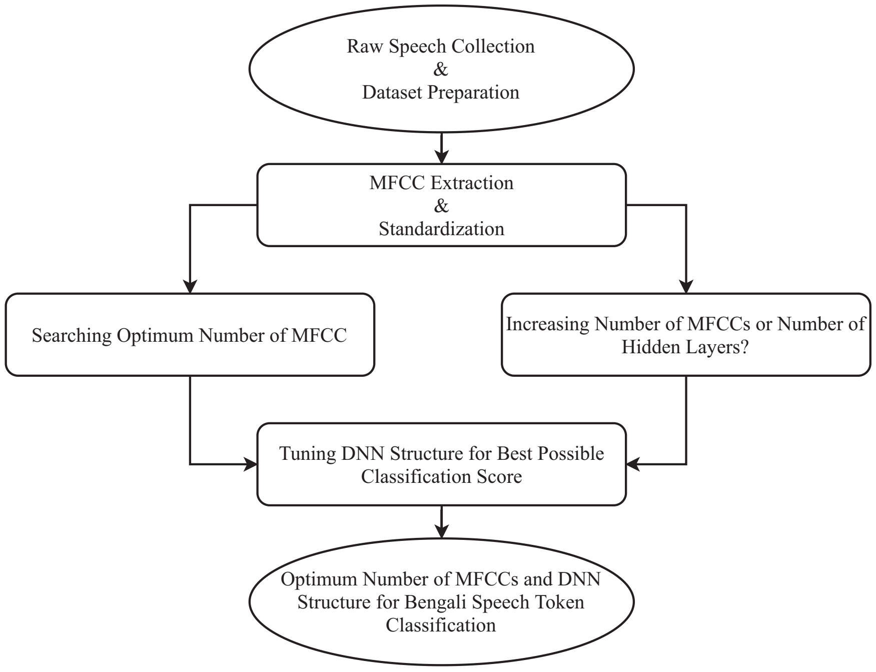

Raw Speech Collection & Dataset Preparation
MFCC Extraction & Standardization
Searching Optimum Number of MFCC
Increasing Number of MFCCs or Number of Hidden Layers?
Tuning DNN Structure for Best Possible Classification Score
Optimum Number of MFCCs and DNN Structure for Bengali Speech Token Classification

FIGURE 1 Steps involved in this work

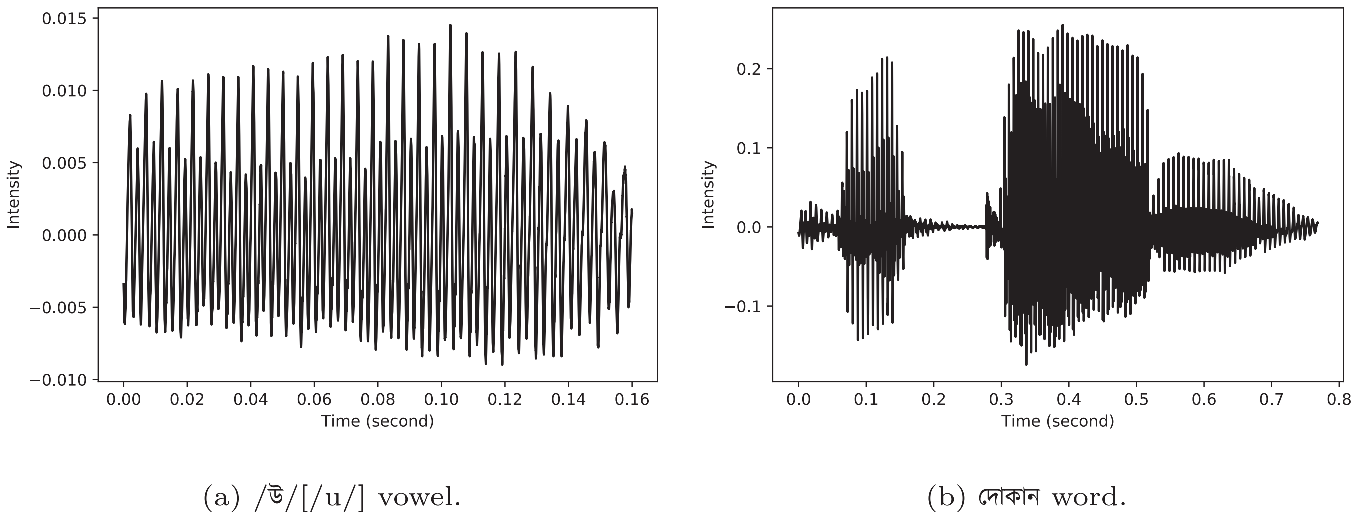


FIGURE 2 Waveshapes of a vowel and a word sound from the datasets

the datasets. It portrays that the vowel waveform is relatively steady, but the word waveform has more variations as it is composed of consonants and vowels.

The volume of the datasets used in this work is comparable to other similar researches, such as [18], [31], and [32]. Particularly for Bengali voice recognition, Syfullah et al. [18] used only 20 different sample inputs for each of their Bengali characters to classify. In a neural network-based word classification, Selvan and Rajesh [31] used 42 samples for each of their words. On the contrary, our work has utilized 40 different samples for each of our Bengali characters and words. Furthermore, Baquirin and Fernandez [32] showed that 110 sound clips are reasonable for this type of works, whereas we used 280 sound clips for both vowel and word classification.

2.2 | MFCC extraction and standardization

MFCC feature extraction process involves applying discrete Fourier transform on a signal window, taking the logarithm, and then expressing on a Mel scale, followed by a discrete cosine transform (DCT) [9]. Then the DCT components refer to the MFCCs. We have utilized version 0.7.2 of the librosa package [33] in the Python programming language to extract the MFCC features. Extracted MFCC features have a different range of values, and to utilize these values with a target to converging the DNN model faster, we have standardized all MFCCs [34] according to Equation (1).

$$x_i' = \frac{x_i - \overline{x_i}}{\sigma_i} \tag{1}$$

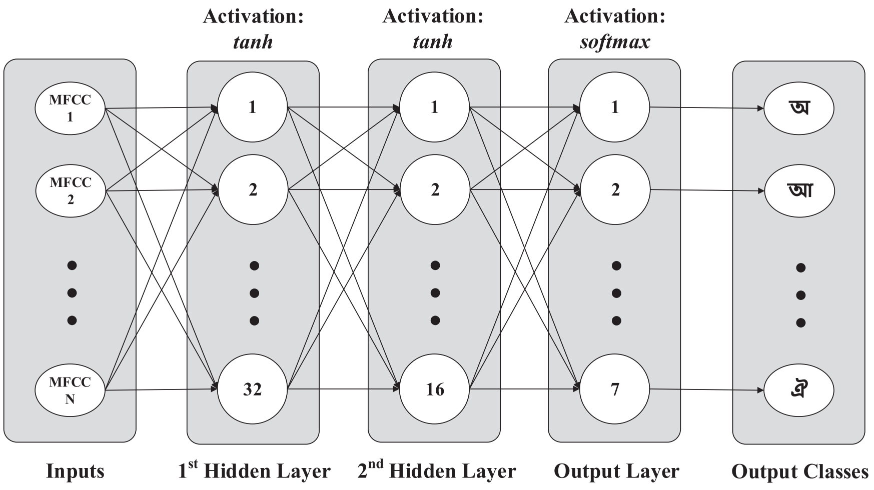


FIGURE 3 The architecture of a fully connected DNN-based vowel classification model having two hidden layers. Input neurons represent the MFCC features, and output neurons represent the output vowel classes of the vowel classification model

where x_i is the i^{th} MFCC, $\overline{x_c}$ and σ_i are the mean and standard deviation of i^{th} MFCC. After that, we utlize these standardized MFCC x_i' in classification of vowels and words.

2.3 | Deep neural network model

In a typical deep neural network-based classifier, the number of neurons in the input layer equals the number of input features used, and the number of neurons in the output layer equals the number of output classes to classify. In our case, we have seven output classes for both vowel and word classification models, and we varied the number of input MFCC features to find out the optimum number of MFCC features. There could be several hidden layers in between the input and output layers depending on the classification scenario. We have utilized two hidden layers of 32 and 16 neurons, respectively, to find out the optimum number of MFCC, the architecture of which is presented in Figure 3 particularly for vowel classification. As we varied the number of input features during the optimum number of MFCC searching, the input neurons of the Figure 3 changed accordingly. Plus, for word classification, the output classes correspond to seven words. Furthermore, after finding the optimum number of MFCC features, we tried to obtain the best possible score, and to do this, we varied the number of hidden layers and the number of neurons on them.

2.3.1 | Weights, biases, and hyperparameters

Weights and biases are two fundamental trainable parameters of any DNN-based model. In any dense layer, the output of any neuron is calculated according to Equation (2).

$$v = b + \sum_{i=1}^{n} x_i \cdot w_i \qquad (2)$$
$$y = f(v)$$

where f = activation function, b = bias, w = weights, x = input to neuron, n = the number of inputs from the incoming layer, and v refers to the linear operation that is applied to each and every neurons in the model. After the linear operation, an activation function is applied to all neurons in a layer to adapt to the system's non-linearity and restrict the output values within a certain limit defined by that particular activation function type. In hidden layers, we have chosen *tanh* activation function since it gave us better performance while experimenting with several activation functions such as *ReLU* and *Leaky ReLU*. The *tanh* activation limits the output from −1 to +1 as shown in Equation (3).

$$f(v) = \tanh(v) = \frac{e^v - e^{-v}}{e^v + e^{-v}} \qquad (3)$$

For a multi-class classification model like what we are dealing with, *Softmax* activation function is an optimal choice in the output layer. It returns the probabilities of each class through which the target class is determined, having the highest probability close to one [35]. The mathematical equation is given in Equation (4).

$$\hat{y}_j = f(v_j) = \frac{e^{v_j}}{\sum_{j=1}^{K} e^{v_j}}; \quad j = 1, 2, 3, \ldots K \qquad (4)$$

where K is the total number of output classes, and $\hat{y}_j$ denotes the prediction for j^{th} class.

One pass of the full training set through the model is termed as one *epoch*. We divided the whole dataset for both vowel and word into a training set of 80% and a validation set of 20%. In each *epoch*, all of these trainable parameters are slightly changed towards the best values by minimizing the loss function. This loss function is actually a measure of the difference between the actual output and predicted output. We have used *categorical cross-entropy* loss function as defined by Equation (5).

$$\text{Loss} = -\sum_{j=1}^{K} y_j \cdot \log \hat{y}_j \qquad (5)$$

where y_j is the ground-truth or target value, and $\hat{y}_j$ denotes the prediction made by the model for j^{th} class. More deviation between these two will result in a higher loss score. The key technique behind optimizing a DNN is to use this score as a feedback signal to fine-tune the trainable parameters gradually to reduce this loss score. An optimizer makes this adjustment through which it implements the fundamental *Backpropagation* algorithm. The optimizer used in this work is *Adam* with a learning rate of 0.005. It is one of the heavily used optimization algorithms in the deep learning domain [36, 37].

The number of trainable parameters in the DNN model depends on the number of input neurons. Each neuron has a weight parameter and bias parameter corresponding to the preceding layer. For a fully connected network (Figure 3), the number of trainable parameters of a particular layer is defined according to Equation (6).

$$TP_l = n_l \times n_{l-1} + n_l \tag{6}$$

where n_l refers to the number of neurons on layer l, n_{l-1} refers to the number of neurons on the preceding layer of l, and TP_l refers to the total trainable parameters at layer l. The term $(n_l \times n_{l-1})$ denotes the number of weight parameters, and the last term (n_l) denotes the number of bias parameters.

2.3.2 | Evaluation metrics

To measure the performances of the classification, we utilized five different metrics—classification accuracy, area under curve of receiver operating characteristic (AUC-ROC), F_1 score, Cohen's κ, and confusion matrix. Classification accuracy denotes the correct classification rate for all classes. AUC-ROC is a better evaluation metric, especially when the number of samples at different classes is different. On top of these, some other metrics named Precision is the ratio of the number of correctly classified labels to all predictions the model picked as correct, including those not identified correctly, and Recall is the ratio of the number of correctly classified labels to all labels that should have been classified correctly (i.e. all ground truths). In most cases, an inverse proportional relationship exists between Precision and Recall, and therefore, a harmonic mean of these two metrics gives a better estimate of the model's performance. This metric is known as F_1 score or F-measure which is calculated according to Equation (7).

$$F_1 \text{ score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7}$$

Cohen's κ is a statistical metric that represents the performance of the classifier other than arbitrary assumption [38]. According to Landis and Koch [39], a value of less than zero tells that the model is giving a random guess; a value of 0.81 to 1.00 implies an almost perfect model; a value of 0.61 to 0.8 indicates a substantial-good model, and so on. For all these four evaluation metrics or accuracy indexes, a higher value indicates better performance. Lastly, the confusion matrix tells us the classification performance in terms of individual classes, and it is related to Precision, Recall, and F_1 score.

Different performance metrics have different pros and cons, and a model's performance should not be justified based on a single metric. That is why we have considered all these five performance metrics. We have utilized the K-fold cross-validation technique in which the total dataset is split into K number of folds. In each run, (K−1) folds are used to train the model, and the remaining single fold is used to validate the model. Thus, K runs are required to traverse all the folds in total, and then we use the average performance of all of these folds as a cross-validated evaluation metric.

2.4 | Overall workflow

The key steps involved in this work are mentioned below.

1. The datasets presented in Section 2.1 (Speech collection and dataset preparation) are processed according to Section 2.2 (MFCC extraction and standardization), through which we extracted standardized MFCC features.
2. The standard features are then fed to the classification model described in Section 2.3 (Deep neural network model). We performed the following three experiments:
   (a) To seek the optimum number of MFCCs, we gradually increased the numbers of MFCC features from 8 to 28 for both vowel and word classification in the two-hidden-layered network presented in Figure 3.
   (b) As we have already discussed in Section 1 (Introduction) that both increasing the number of MFCCs and hidden layers raises the computational burden while enhancing the classification performance, this research also aims to find which increment is more suitable. For this purpose, we have increased the number of hidden layers and calculated the classification metrics. Then both metrics (obtained by increasing the number of MFCCs and hidden layers) are compared based on the number of trainable parameters.
   (c) Finally, to elicit the best possible score, we have utilized the optimum number of MFCCs found in item 2a and then experimented by increasing the number of hidden layers and the number of neurons in them.

3 | RESULTS AND DISCUSSION

This section reports and explains the findings of this article. In general, we first seek the optimum number of MFCC, then we analyze between hidden layer increase and MFCC increase, and

TABLE 1 Performance comparison with respect to the variation of the number of MFCC features for both vowel and word classification

Type	MFCCs	Parameters	Accuracy	AUC-ROC	F_1 score	Cohen's κ
Vowel	8	935	0.66 (± 0.01)	0.94 (± 0.00)	0.65 (± 0.01)	0.61 (± 0.01)
	10	999	0.70 (± 0.01)	0.95 (± 0.00)	0.68 (± 0.01)	0.64 (± 0.01)
	12	1063	0.71 (± 0.01)	0.95 (± 0.01)	0.71 (± 0.01)	0.67 (± 0.01)
	13	1095	0.74 (± 0.02)	0.96 (± 0.00)	0.73 (± 0.02)	0.69 (± 0.02)
	14	1127	0.75 (± 0.02)	0.96 (± 0.00)	0.74 (± 0.02)	0.71 (± 0.02)
	16	1191	0.78 (± 0.01)	0.97 (± 0.00)	0.78 (± 0.01)	0.74 (± 0.01)
	18	1255	0.80 (± 0.00)	0.97 (± 0.00)	0.80 (± 0.00)	0.77 (± 0.00)
	20	1319	0.82 (± 0.01)	0.98 (± 0.00)	0.82 (± 0.01)	0.79 (± 0.01)
	22	1383	0.83 (± 0.01)	0.98 (± 0.00)	0.82 (± 0.01)	0.80 (± 0.01)
	24	1447	0.84 (± 0.00)	0.98 (± 0.00)	0.84 (± 0.00)	0.81 (± 0.01)
	25	**1479**	**0.83 (± 0.01)**	**0.98 (± 0.00)**	**0.83 (± 0.01)**	**0.81 (± 0.01)**
	26	1511	0.83 (± 0.01)	0.98 (± 0.00)	0.82 (± 0.01)	0.80 (± 0.01)
	27	1543	0.84 (± 0.00)	0.98 (± 0.00)	0.84 (± 0.00)	0.81 (± 0.00)
	28	1575	0.84 (± 0.00)	0.98 (± 0.00)	0.84 (± 0.00)	0.82 (± 0.00)
Word	8	935	0.46 (± 0.00)	0.83 (± 0.00)	0.44 (± 0.00)	0.37 (± 0.00)
	10	999	0.47 (± 0.01)	0.84 (± 0.01)	0.45 (± 0.01)	0.38 (± 0.01)
	12	1063	0.49 (± 0.01)	0.85 (± 0.00)	0.47 (± 0.01)	0.40 (± 0.01)
	13	1095	0.51 (± 0.01)	0.86 (± 0.00)	0.49 (± 0.01)	0.42 (± 0.01)
	14	1127	0.52 (± 0.01)	0.86 (± 0.00)	0.51 (± 0.01)	0.44 (± 0.01)
	16	1191	0.51 (± 0.00)	0.86 (± 0.00)	0.50 (± 0.00)	0.43 (± 0.01)
	18	1255	0.53 (± 0.01)	0.87 (± 0.00)	0.52 (± 0.01)	0.45 (± 0.01)
	20	1319	0.54 (± 0.01)	0.87 (± 0.00)	0.52 (± 0.01)	0.46 (± 0.01)
	22	1383	0.55 (± 0.00)	0.88 (± 0.00)	0.54 (± 0.00)	0.47 (± 0.00)
	24	1447	0.56 (± 0.01)	0.88 (± 0.00)	0.55 (± 0.01)	0.48 (± 0.01)
	25	**1479**	**0.57 (± 0.01)**	**0.88 (± 0.00)**	**0.56 (± 0.01)**	**0.49 (± 0.01)**
	26	1511	0.56 (± 0.01)	0.88 (± 0.00)	0.55 (± 0.01)	0.48 (± 0.01)
	27	1543	0.57 (± 0.01)	0.88 (± 0.00)	0.56 (± 0.01)	0.49 (± 0.01)
	28	1575	0.56 (± 0.00)	0.88 (± 0.00)	0.56 (± 0.00)	0.49 (± 0.00)

Note: The parameters represent the total number of trainable parameters. It denotes that twenty-five MFCCs seems optimum for both vowel and word classification. For this whole comparison, we trained the model for 50 epochs in the DNN configuration of two hidden layers shown in Figure 3 with all random initialization fixed to a seed value of 42.

finally, we seek the best classification performance, followed by a comparison with related studies.

3.1 | Search for optimum number of MFCC

The architecture depicted in Figure 3 is utilized for demonstrating performances for a varying number of MFCC for both vowel and word classification. Table 1 reports a detailed comparison with respect to the variation in the number of MFCC features for both vowel and the word classification.

Since the performance of the model may vary at different runtimes, we utilized four-fold cross-validation, and the average scores of these four runtimes are shown in Table 1 along with the standard deviation in parentheses. While comparing, all random initializations are fixed from a constant seed value of 42. This technique ensures the reproducibility of identical performance scores in multiple runtimes. For this comparison table, the two-hidden-layered network was trained for 50 epochs.

For both vowel and word classification, the performance scores increase with an increasing number of MFCCs. For vowel classification, they were highest for 24 MFCCs in input, whereas for word classification, the highest scores of the same metrics were observed for 25 MFCCs in input. From 13 MFCCs to 25 MFCCs, vowel shows $0.83 - 0.74 = 0.09$ that is 9%, and word shows $0.57 - 0.51 = 0.06$ that is 6% increase in overall accuracy. It is seen from Table 1 that 24 MFCCs takes less parameters and provides better accuracy and F_1 score for vowels, and provides lower accuracy, F_1 score, and Cohen's κ for words compared to 25 MFCCs, although the difference is much smaller. One can use either 24 or 25 MFCCs for their evaluation, but we choose 25 MFCCs for being consistent.

TABLE 2 Performance comparison with increasing the number of hidden layers at 13 MFCCs. The model was cross-validated (four-fold) and trained for 50 epochs

Type	Model	Parameters	Time (s)[c]	Accuracy	AUC-ROC	F_1 score	Cohen's κ
Vowel	HL_2[a]	1095	5.40	0.74 (± 0.02)	0.96 (± 0.00)	0.73 (± 0.02)	0.69 (± 0.02)
	HL_3[b]	3623	5.88	0.79 (± 0.01)	0.97 (± 0.00)	0.79 (± 0.01)	0.76 (± 0.01)
Word	HL_2	1095	5.83	0.51 (± 0.01)	0.86 (± 0.00)	0.49 (± 0.01)	0.42 (± 0.01)
	HL_3	3623	6.29	0.55 (± 0.01)	0.89 (± 0.00)	0.55 (± 0.01)	0.48 (± 0.01)

[a]HL_2: Two hidden layers having 32-16 neurons
[b]HL_3: Three hidden layers having 64-32-16 neurons
[c]Time (s): The execution time (in seconds) of both the training and validation phase together.

3.2 | MFCC increase versus hidden layer increase

It is apparent that increasing the number of hidden layers should also increase the performance, and that is why we performed the same classification experiment with three hidden layers (Table 2), and as expected, the performance increased. However, the use of three hidden layers increases the number of trainable parameters, thereby increasing computational burden, which is a crucial limit in implementing speech recognition systems in end devices, such as microcontrollers and Arduino. The table also reports total execution time (both training and validation phase together) in an HP Pavilion 14 laptop with 64-bit Linux Mint 19.3 OS, 8 GB RAM, 1.7 GHz Intel Core i5-4210U Processor, and NVIDIA GM108M [GeForce 830M] 2 GB Graphics.

As shown in Table 2, for vowel classification employing 13 MFCCs, two hidden layers to three hidden layers introduce 5% increase in accuracy, 1% increase in AUC-ROC, 6% increase in F_1 score, and 7% increase in Cohen's κ. At the same time, these performance increases require $3623 - 1095 = 2528$ additional trainable parameters. Similar increases of performances in three hidden layers is comparable to 16 or 17 MFCCs of Table 1. Therefore, if we increase the number of MFCCs to increase the performances, it would cost only $1191 - 1095 = 96$ additional trainable parameters. As compared to execution time, 16 MFCCs required only 5.53 s which is also less than the execution time in three-hidden-layered configuration (HL_3). Comparing these additional parameters in two cases, we can comment that increasing MFCCs is computationally efficient than increasing hidden layers since more trainable parameters require more time and computational power to do the classification. Word classification also depicts similar results. An increase of 4% classification accuracy can be done either by additional 2528 parameters if we want to increase the number of hidden layers or by additional $1383 - 1095 = 288$ parameters if we want to increase the number of MFCCs to 22. As for execution time, 22 MFCCs took only 5.75 s, less than what it took for HL_3. Accordingly, this guides us to increase the number of MFCCs rather than the number of hidden layers. The analysis further suggests that the optimum number of MFCCs could be 24 or 25, although the optimum number of MFCCs is well known to be 13.

TABLE 3 Best performance scores with increasing the number of hidden layers at 25 MFCCs. The evaluation metrics are four-fold cross-validated

Type	Model	Accuracy	AUC-ROC	F_1 score	Cohen's κ
Vowel	HL_2[a]	0.96 (± 0.00)	1.00 (± 0.00)	0.96 (± 0.00)	0.95 (± 0.00)
	HL_4[b]	0.99 (± 0.00)	1.00 (± 0.00)	0.99 (± 0.00)	0.98 (± 0.00)
	HL_5[c]	**0.99 (± 0.00)**	**1.00 (± 0.00)**	**0.99 (± 0.00)**	**0.98 (± 0.00)**
Word	HL_2	0.75 (± 0.01)	0.96 (± 0.00)	0.75 (± 0.01)	0.71 (± 0.01)
	HL_4	0.89 (± 0.01)	0.98 (± 0.00)	0.89 (± 0.01)	0.87 (± 0.01)
	HL_5	**0.91 (± 0.01)**	**0.98 (± 0.00)**	**0.91 (± 0.01)**	**0.90 (± 0.01)**

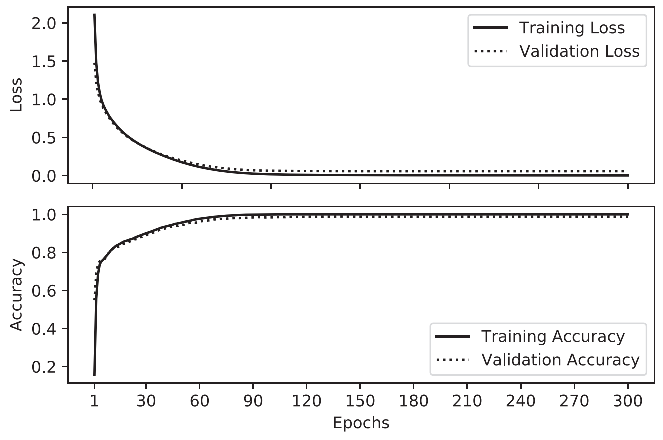
[a]HL_2: Two hidden layers having 32-16 neurons
[b]HL_4: Four hidden layers having 128-64-32-16 neurons
[c]HL_5: Five hidden layers having 128-128-64-32-16 neurons

In a search for the optimum number of MFCCs in English and Portuguese digit classification, Silva et al. [28] report a decrease in performance after 25 MFCCs, and they concluded that the optimum number of coefficients has a narrow range between 11 to 23 MFCCs. Our findings also comply with it. The authors in [12] also experimented by varying the number of MFCCs and other parameters to find the best configuration for a low-resource embedded system. They experimented with 8, 9, and 10 MFCCs, and concluded that 9 MFCCS are suitable in accordance with other parameter tuning such as the number of filters and number of HMM states. A significant reason behind choosing nine MFCCs was less training and recognition time, particularly for low-resource embedded systems they considered.

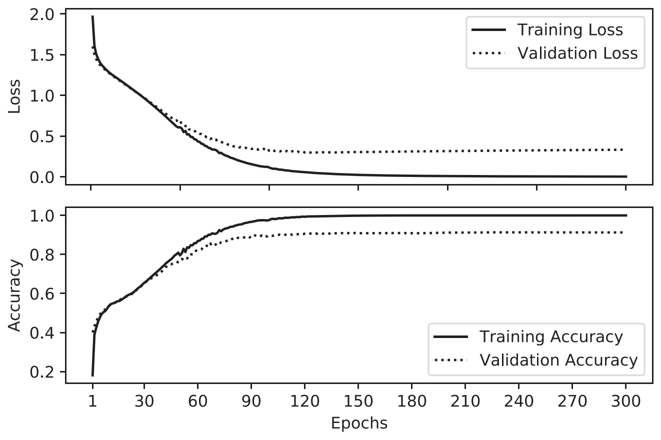
3.3 | Search for best scores

The classification performances have a proportional relationship with the number of MFCCs and the number of hidden layers. According to the previous discussions (Section 3.2), for enhancing the classification score, we have to first concentrate on incorporating the optimum number of MFCC, and then, we should extend the number of hidden layers. As we already concluded that 25 MFCCs would be the optimum number of MFCCs, we increased the number of hidden layers while utilizing 25 MFCCs shown in Table 3.

While increasing the number of hidden layers, we found five hidden layers as optimum since performance scores did not

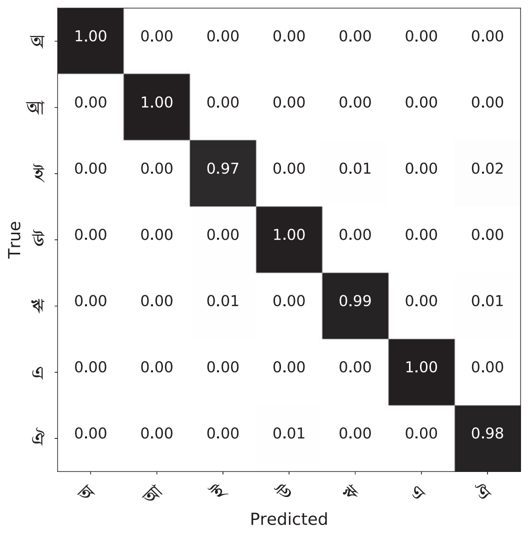


(a) Vowel classification.

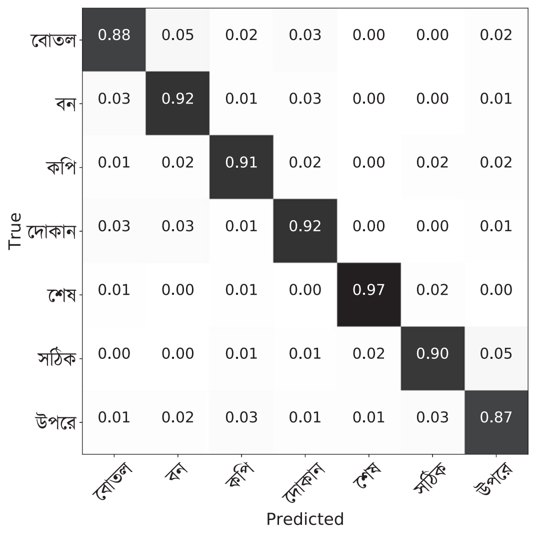


(b) Word classification.

FIGURE 4 Loss minimization and accuracy score during training and validating using MFCC features. The classification model consists of five hidden layers



(a) Vowel classification.



(b) Word classification.

FIGURE 5 Confusion matrices for the vowel and the word classification. It indicate the classification performances of individual vowels and words

increase for six and more hidden layer configurations. We also checked with a varying number of hidden neurons in the configurations shown in Table 3, but the reported configurations were found optimum. Finally, we trained the model for 300 epochs to extract the best possible scores, which resulted in 99% and 91% classification accuracies for vowels and words, respectively.

For the best scores, Figures 4a and 4b depict the loss minimization and accuracy score curves during both training and validation phases for both vowel and word classification, respectively.

Since there is no large difference between training and validation curves for vowel classification, the model is not overfitting. The smoother curves for vowel classification (Figure 4a) indicate proper tuning of model hyperparameters. It further means that the formation of the DNN classifier is good enough to have such a good model. However, for word classification, there are some differences between training and validation losses shown in Figure 4b. A possible reason behind this is the presence of more dynamic acoustical features in words [40].

Classification performance for individual labels can be identified from the confusion matrices shown in Figure 5. The top-left to bottom-right diagonal elements in the matrix represent the accurate classification rate for respective classes.

As shown from the matrix, vowel classification was perfect with 100% accuracy for /অ/[/অ/], /আ/[/a/], /উ/[/u/], and /এ/[/e/] vowels. The class that had the lowest accuracy with this model was /ই/[/i/], and it was 97% correct. On the contrary, the model's best performance for the word classification

TABLE 4 Comparison with relevant studies based on the used number of MFCCs and reported classification accuracy. The comparison proves the competitiveness of our classification model with 25 MFCCs, whereas the general wisdom is to use 13 MFCCs

Model	Article	Recognition Domain	MFCCs	Accuracy
DNN	**Proposed**	**Seven Bengali vowels**	**25**	**99%**
	[24]	Three Arabic words	13	92.42%
	Proposed	**Seven Bengali words**	**25**	**91%**
	[21]	English phonemes	Not specified	90.77%
	[25]	English command words	13	82.46%
	[18]	Bengali speech characters	Not specified	81.61%
	[22]	Malayalam Vowels	12	74.39%
CNN	[13]	10 spoken Bengali digits	Not specified	98.37%
	[26]	Isolated English words	Not specified	96.19%
	[7]	Nigerian accent classification	40	94.9%
	[40]	Isolated Bengali vowels	13	93.93%
	[40]	Isolated Bengali words	13	90 %
	[23]	Dari speech tokens	13	88.2%
	[16]	Isolated Bengali speech	Not specified	86.058%
	[17]	Ten Bengali short speech words	13	74.01%
SVM	[27]	Speech emotion	13, Δ, $\Delta-\Delta$	90%
CMUSphinx	[14]	Bengali word	13, Δ, $\Delta-\Delta$	85.3%
	[15]	Bengali speech	13, Δ, $\Delta-\Delta$	Not specified
HTK	[14]	Bengali phoneme	13, Δ, $\Delta-\Delta$	54.07%

CMUSphinx: Open source speech recognition system developed at Carnegie Mellon University
CNN: Convolutional neural network
DNN: Deep neural network
HTK: Hidden Markov model toolkit
SVM: Support vector machine
Δ: First derivatives of 13 base MFCCs
$\Delta-\Delta$: Second derivatives of 13 base MFCCs

was for the শেষ word, and it was 97% correct. For all other words, the model's performance was quite satisfactory with an 87% classification rate as the lowest found for উপরে word.

3.4 | Comparison with other studies

Our main target was to investigate the number of MFCC features required to achieve the best classification performance. Table 4 reports some relevant studies, where at first, we present the DNN-based studies, then the CNN-based studies followed by other approaches. In each category, we ordered the articles according to the highest classification performance to the lowest.

Table 4 reveals that most of the speech classification researches employ either DNN or CNN, which is the extended version of DNN architecture added with convolutional layers. Several studies also utilize the first and second derivatives of 13 base MFCCs to consider the temporal dynamics of the speech signal [9]. Since the recognition domain and datasets are different, we should not strictly compare the classification performance among these studies. However, competitive classification performance proves the applicability and suitability of our model. There are two concrete outcomes from this table—most articles utilized only 13 base MFCCs where our research recommends 24 or 25 MFCCs, and our classification accuracy outperforms state-of-the-art scores on vowel recognition. Also, our word classification accuracy is competitive to similar studies.

4 | CONCLUSION

Speech is a significant natural source of information used in many aspects—speech dictation gadgets, accent classification, emotion recognition, and disease diagnosis, to name a few. Therefore, speech-related researches have a meaningful impact on our day-to-day life. Since audio data cannot be processed directly in many cases, researchers extract valuable information from the audio, what we call feature extraction. Among many features, MFCC has been widely used by researchers all over the world. Although it is a general wisdom to use the first 13 coefficients, we put a test to answer the question—how many MFCCs are to be utilized? We performed the classification of seven Bengali vowels and seven Bengali words by varying MFCC numbers from 8 to 28. In a two-hidden-layered DNN model, 13 MFCCs gave 74% vowel and 51% word classification accuracy, whereas 25 MFCCs gave 83% vowel and 57% word classification accuracy, from which we recommend that 25 MFCCs could be the optimum number of MFCCs. In fact, the general wisdom of 13 MFCCs could serve the same performance score if we increase the number of DNN hidden layers. However, it increases the total trainable parameters (computational burden), a limiting factor to implement speech recognition systems in edge devices like Arduino. With the optimum number of MFCCs discovered in this study, we further seek the best possible scores by increasing the hidden layers. Accordingly, in a five-hidden-layered model, we obtain 99% vowel and 91% word classification accuracy that is competitive to other similar speech classification studies. The outcome of this study will be helpful for future MFCC-based researches. Employing the optimum number of MFCCs should increase the overall performance of devices, systems, and research involving MFCC.

CONFLICT OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in Mendeley Data at https://doi.org/10.17632/2h6975kdsx.1.

ORCID

Md. Rakibul Hasan https://orcid.org/0000-0003-2565-5321
Md. Mahbub Hasan https://orcid.org/0000-0003-2612-7248
Md Zakir Hossain https://orcid.org/0000-0003-1892-831X

REFERENCES

1. Hemmerling, D., Wojcik-Pedziwiatr, M.: Prediction and estimation of Parkinson's disease severity based on voice signal. J. Voice (2020). https://doi.org/10.1016/j.jvoice.2020.06.004 In press
2. Mirheidari, B., Blackburn, D., Walker, T., Reuber, M., Christensen, H.: Dementia detection using automatic analysis of conversations. Comput. Speech & Lang. 53, 65–79 (2019). https://doi.org/10.1016/j.csl.2018.07.006
3. Gosztolya, G., Vincze, V., Tóth, L., Pákáski, M., Kálmán, J., Hoffmann, I.: Identifying mild cognitive impairment and mild alzheimer's disease based on spontaneous speech using asr and linguistic features. Comput. Speech & Lang. 53, 181–197 (2019). https://doi.org/10.1016/j.csl.2018.07.007
4. Yağanoğlu, M.: Real time wearable speech recognition system for deaf persons. Comput. & Electr. Eng. 91, 107026 (2021). https://doi.org/10.1016/j.compeleceng.2021.107026
5. Veiga, A., Lopes, C., Sá, L., Perdigão, F.: Acoustic similarity scores for keyword spotting. In: Baptista, J., Mamede, N., Candeias, S., Paraboni, I., Pardo, T.A.S., Volpe Nunes, M.d.G. (eds.) Computational Processing of the Portuguese Language, pp. 48–58. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-09761-9_5
6. Shahriar, S. & Kim, Y.: Audio-visual emotion forecasting: Characterizing and predicting future emotion using deep learning. In: 2019 14th IEEE International Conference on Automatic Face Gesture Recognition (FG 2019), pp. 1–7. Lille (2019). https://doi.org/10.1109/FG.2019.8756599
7. Salau, A.O., Olowoyo, T.D., Akinola, S.O.: Accent classification of the three major Nigerian indigenous languages using 1D CNN LSTM network model. In: Jain, S., Sood, M., Paul, S. (eds.) Advances in Computational Intelligence Techniques, pp. 1–16. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-2620-6_1
8. Suresh, M.J.S. & Thorat, S.: Language identification system using MFCC and SDC feature. In: Proceedings of 4th RIT Post Graduates Conference (RIT PG Con-18), pp. 113–119. Islampur (2018)
9. Rao, K.S., Manjunath, K.E.: Speech recognition using articulatory and excitation source features. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-49220-9
10. Sultana, S., Rahman, M.S., Iqbal, M.Z.: Recent advancement in speech recognition for bangla: A survey. Int. J. Adv. Comput. Sci. Appl. 12(3), 546–552 (2021). https://doi.org/10.14569/IJACSA.2021.0120365
11. Al-Anzi, F. & AbuZeina, D.: Literature survey of arabic speech recognition. In: 2018 International Conference on Computing Sciences and Engineering (ICCSE), pp. 1–6. Kuwait (2018). https://doi.org/10.1109/ICCSE1.2018.8374215
12. da Silva, D.D.C., Vasconcelos, C.R., Neto, B.G.A. & Fechine, J.M.: Evaluation of the impact in reducing the number of parameters for continuous speech recognition for brazilian portuguese. In: 2012 ISSNIP Biosignals and Biorobotics Conference: Biosignals and Robotics for Better and Safer Living (BRC), pp. 1–6. Manaus (2012). https://doi.org/10.1109/BRC.2012.6222182
13. Sharmin, R., Rahut, S.K., Huq, M.R.: Bengali spoken digit classification: A deep learning approach using convolutional neural network. Procedia Comput. Sci. 171, 1381–1388 (2020). https://doi.org/10.1016/j.procs.2020.04.148
14. Das, B., Mandal, S. & Mitra, P.: Bengali speech corpus for continuous auutomatic speech recognition system. In: 2011 International Conference on Speech Database and Assessments (Oriental COCOSDA), pp. 51–55. Hsinchu (2011). https://doi.org/10.1109/ICSDA.2011.6085979
15. Mandal, S., Das, B. & Mitra, P.: Shruti-ii: A vernacular speech recognition system in Bengali and an application for visually impaired community. In: 2010 IEEE Students Technology Symposium (TechSym), pp. 229–233. Kharagpur (2010). https://doi.org/10.1109/TECHSYM.2010.5469156
16. Islam, J., Mubassira, M., Islam, M.R. & Das, A.K.: A speech recognition system for Bengali language using recurrent neural network. In: 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), pp. 73–76. Singapore (2019). https://doi.org/10.1109/CCOMS.2019.8821629
17. Sumon, S.A., Chowdhury, J., Debnath, S., Mohammed, N. & Momen, S.: Bangla short speech commands recognition using convolutional neural networks. In: 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1–6. Sylhet (2018). https://doi.org/10.1109/ICBSLP.2018.8554395
18. Syfullah, S.M., Zakaria, Z.B., Uddin, M.P., Rabbi, M.F., Afjal, M.I. & Nitu, A.M.: Efficient vector code-book generation using k-means and linde-buzo-gray (lbg) algorithm for Bengali voice recognition. In: 2018 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE), pp. 1–4. Gazipur (2018). https://doi.org/10.1109/ICAEEE.2018.8642994
19. Badhon, S.M.S.I., Rahaman, M.H., Rupon, F.R. & Abujar, S.: State of art research in Bengali speech recognition. In: 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), pp. 1–6. Kharagpur (2020). https://doi.org/10.1109/ICCCNT49239.2020.9225650
20. Devi, K.J., Singh, N.H., Thongam, K.: Automatic speaker recognition from speech signals using self organizing feature map and hybrid neural network. Microprocessors Microsys. 79(4), 103264 (2020). https://doi.org/10.1016/j.micpro.2020.103264
21. Bird, J.J., Wanner, E., Ekárt, A., Faria, D.R.: Optimisation of phonetic aware speech recognition through multi-objective evolutionary algorithms. Expert Syst. Appl. 153, 113402 (2020). https://doi.org/10.1016/j.eswa.2020.113402
22. Mohamed, F.K., Lajish, V.: Nonlinear speech analysis and modeling for Malayalam vowel recognition. Procedia Comput. Sci. 93, 676–682 (2016). https://doi.org/10.1016/j.procs.2016.07.261
23. Dawodi, M., Baktash, J.A., Wada, T., Alam, N. & Joya, M.Z.: Dari speech classification using deep convolutional neural network. In: 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), pp. 1–4. Vancouver (2020). https://doi.org/10.1109/IEMTRONICS51293.2020.9216370

24. Wahyuni, E.S.: Arabic speech recognition using mfcc feature extraction and ann classification. In: 2017 2nd International conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE), pp. 22–25. Yogyakarta (2017). https://doi.org/10.1109/ICITISEE.2017.8285499
25. Yang, X., Yu, H. & Jia, L.: Speech recognition of command words based on convolutional neural network. In: 2020 International Conference on Computer Information and Big Data Applications (CIBDA), pp. 465–469. Guiyang (2020). https://doi.org/10.1109/CIBDA50819.2020.00110
26. Soliman, A., Mohamed, S. & Abdelrahman, I.A.: Isolated word speech recognition using convolutional neural network. In: 2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE), pp. 1–6. Khartoum (2021). https://doi.org/10.1109/ICCCEEE49695.2021.9429684
27. Cen, L., Wu, F., Yu, Z.L., Hu, F.: Chapter 2 - a real-time speech emotion recognition system and its application in online learning. In: Tettegah, S.Y., Gartmeier, M. (eds.) Emotions, Technology, Design, and Learning, Emotions and Technology, pp. 27–46. Academic Press, San Diego (2016). https://doi.org/10.1016/B978-0-12-801856-9.00002-5
28. Silva, D.F., Souza, V.M.A.d., Batista, G.E.A.P.A.: A comparative study between MFCC and LSF coefficients in automatic recognition of isolated digits pronounced in Portuguese and English. Acta Scientiarum Technol. 35(4), 621–628 (2013). https://doi.org/10.4025/actascitechnol.v35i4.19825
29. Hasan, M.R. & Hasan, M.M.: Isolated Bengali vowel and word speech sounds. Mendeley Data v1 (2021). https://doi.org/10.17632/2h6975kdsx.1
30. Audacity Team. Audacity®: Free audio editor and recorder [computer application]. (2018). https://audacityteam.org/
31. Selvan, A.M., Rajesh, R.: Word classification using neural network. In: Abraham, A., Mauri, J.L., Buford, J.F., Suzuki, J., Thampi, S.M. (eds.) Advances in Computing and Communications, pp. 497–502. Springer, Berlin (2011). https://doi.org/10.1007/978-3-642-22720-2_52
32. Baquirin, R.B.M., Fernandez, P.L.: Artificial neural network (ANN) in a small dataset to determine neutrality in the pronunciation of english as a foreign language in filipino call center agents: Neutrality classification of Filipino call center agent's pronunciation. Inteligencia Artificial 21(62) 134–144 (2018). https://doi.org/10.4114/intartif.vol21iss62pp134-144
33. McFee, B., Lostanlen, V., McVicar, M., Metsai, A., Balke, S., Thomé, C., et al.: librosa/librosa: 0.7.2. (2020). https://doi.org/10.5281/zenodo.3606573
34. Grus, J.: Data Science from Scratch: First Principles with Python, 1st ed. O'Reilly Media, Sebastopol (2019)
35. Chollet, F.: Deep learning with Python, 1st ed. Manning Publications, New York (2018)
36. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
37. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016). http://www.deeplearningbook.org
38. Cohen, J.: A coefficient of agreement for nominal scales. Educational Psychological Measurement 20(1), 37–46 (1960). https://doi.org/10.1177/001316446002000104
39. Landis, J.R., Koch, G.G.: The measurement of observer agreement for categorical data. Biometrics 33(1), 159–174 (1977). http://www.jstor.org/stable/2529310
40. Hasan, M.R. & Hasan, M.M.: Investigation of the effect of MFCC variation on the convolutional neural network-based speech classification. In: 2020 IEEE Region 10 Symposium (TENSYMP), pp. 1408–1411. Dhaka (2020). https://doi.org/10.1109/TENSYMP50017.2020.9230697

How to cite this article: Hasan, M.R., Hasan, M.M., Hossain, M.Z.: How many Mel-frequency cepstral coefficients to be utilized in speech recognition? A study with the Bengali language. J. Eng. 2021, 817–827 (2021). https://doi.org/10.1049/tje2.12082