Kothamala - A Bangla Speech Recognition Engine

Senior Design Project

By

SHAKIL AHMED SUMON	ID# 1431049042
JOYDIP CHOWDHURY	ID# 1512033042
SUJIT DEBNATH	ID# 1511378042

Department of Electrical and Computer Engineering
NORTH SOUTH UNIVERSITY

Faculty Advisor

Dr. Nabeel Mohammed
Assistant Professor | Department of ECE
North South University

FALL 2018

Letter of Transmittal

January, 2019

To

Dr. Shazzad Hosain
Associate Professor and Chairman,
Department of Electrical and Computer Engineering,
North South University,
Bashundhara R/A, Dhaka

Subject: Submission of Capstone Project Report on "Kothamala - A Bangla Speech Recognition Engine".

Dear Sir,

With due respect, we would like to submit Our Capstone Project Report on "Kothamala - A Bangla Speech Recognition Engine" as a part of our BSc program. The report deals with a Bengali transcription system which allows users to transcript their speech into text through an audio input. We tried our level best to make the report meaningful and informative.

The Capstone project was very much valuable to us as it helped us to gain experience from practical field. It was a great learning experience for us. We tried to the maximum competence to meet all the dimensions required from this report.

We will be highly obliged if you are kind enough to receive this report and provide your valuable judgment. It would be our immense pleasure if you find this report useful and informative to have an apparent perspective on the issue.

Sincerely Yours,

Shakil Ahmed Sumon, Joydip Chowdhury, and Sujit Debnath

Department of ECE, North South University,
Bashundhara R/A, Dhaka.

APPROVAL

The capstone project entitled **"Kothamala - A Bangla Speech Recognition Engine"** by Shakil Ahmed Sumon (ID# 1431049042), Joydip Chowdhury (ID# 1512033042) and Sujit Debnath (ID# 1511378042), is approved in partial fulfillment of the requirement of the Degree of Bachelor of Science in Computer Science and Engineering on June, 2018 and has been accepted as satisfactory.

Supervisor:

.....

Dr. Nabeel Mohammed

Assistant Professor
Department of Electrical and Computer Engineering
North South University
Bashundhara R/A, Dhaka.

Department Chair:

.....

Dr. Shazzad Hosain

Associate Professor and Chairman
Department of Electrical and Computer Engineering
North South University
Bashundhara R/A, Dhaka.

AUTHOR'S DECLARATION

This is our truthful declaration that the **"Capstone Project Report"** we have prepared is not a copy of any **"Capstone Project Report"** previously made by any other team. We also express our honest confirmation in support of the fact that the said **"Capstone Project Report"** has neither been used before to fulfill any other course related purpose nor it will be submitted to any other team or authority in future.

.....

Shakil Ahmed Sumon

Department of Electrical and Computer Engineering
North South University, Bangladesh.

.....

Joydip Chowdhury

Department of Electrical and Computer Engineering
North South University, Bangladesh.

.....

Sujit Debnath

Department of Electrical and Computer Engineering
North South University, Bangladesh.

ACKNOWLEDGEMENT

First of all, we wish to express our gratitude to the Almighty for giving us the strength to perform our responsibilities and complete the report.

The capstone project program is very helpful to bridge the gap between the theoretical knowledge and real life experience as part of Bachelor of Science (BSc) program. This report has been designed to have a practical experience through the theoretical understanding. Furthermore, **North South University** gave us plenty of privileges regarding technical facilities. So, we also like to admire the cooperation of our authority.

We also acknowledge our profound sense of gratitude to all the teachers who have been instrumental for providing us the technical knowledge and moral support to complete the project with full understanding.

We would like to convey our gratitude to our faculty **Dr. Nabeel Mohammed** for his stimulating inspiration, kind guidance, valuable suggestions, sagacious advice and kind cooperation throughout the period of word undertaken, which has been instrumented in the success of our project. At this level of understanding it is often difficult to understand the wide spectrum of knowledge without proper guidance and advice. His suggestions and guidance have made the report a good manner.

We like to express our heartiest gratitude to our family and friends for their moral support to carve out this project.

Abstract

Bangla is one of the widest spoken languages of the world, still is counted among the under-resourced languages; hence very few attempts have been undertaken to digitize this very popular language. In this project, we have developed a dataset of popular Bangla short speech commands. The dataset consists of 50 frequently used Bangla short speech commands each of which has 200 utterances. Moreover, experiments have been done to recognize ten short speech commands and three different convolutional neural network architectures have been proposed. One of the models takes raw audio as input whereas another model takes Mel-frequency cepstral coefficients (MFCC) of the audio signals as inputs. However, the other model leverages transfer learning by pre-training the model with English short speech commands. The results show that the model trained with MFCCs performs better in recognizing the short speech commands with 74.01% accuracy whereas the transfer learning model, not far enough from the MFCC model in terms of accuracy, is consistent in recognizing the commands. Moreover, this study proposes an architecture for Bangla language model and along with the language model it develops a Bangla speech to text engine. Three different model architectures have been experimented for building Bangla speech to text engine. As the audios are sequential inputs, one model has been designed with time distributed dense neural network architecture; another model with long short term (LSTM) architecture combined with convolutional layers to extract features from audios. Additionally, we have trained another very small model in combination with one convolutional and one time distributed layer. Connectionist temporal classification (CTC) has been applied as a loss function in all these models. However, spectrogram has been incorporated as an audio preprocessing strategy along with MFCCs and raw audios. The features extracted by the above mentioned pre-processing methods have been used as inputs to all the designed models sequentially. A dataset released by Google is used to train the models. The results will be reported after rigorous testing on the test set.

TABLE OF CONTENTS

List of Tables

Table **Page**

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Automatic speech recognition (ASR) being the peak of natural language processing has attracted a lot of attention from the giant tech leaders like Amazon, Google, Microsoft, Baidu etc. The applications of automatic speech recognition justify this overwhelming attention. Speech inputs to machines provide numerous advantages over traditional I/O methods. Voice inputs are hands-free, natural, location free, eyes free and fast input medium [26]. It makes creating and reading documents painless and gives elderly and disabled people easier access to technology. Therefore, human efforts to develop an automatic speech recognition engine started even before the invention of computers.

However, digital computers which were enabled with A/D converters accelerated the process in the late fifties [26]. Although most of the speech recognition systems which were developed previously had used hand engineered processing systems; researchers nowadays have found that end-to-end speech recognition systems perform better [18]. Moreover, the recent advancement of computing hardware enables these automatic speech recognition engines to be trained with deep learning. In combination with language models, these end-to-end speech recognizers trained with deep learning are algorithmically simpler and performance wise better in hard speech recognition tasks [18].

In today's world, computational power increases significantly, for which numerous progressive work has been done in the sector of the speech recognition system for different speech corpus. Though, in popular language e.g. English, Mandarin etc, several impres-

sive works have been committed regarding transcription. But significant efforts have yet to be made for under-resourced languages like Bengali.

1.1 Engineering Problem

In this study, combined with a language model we have developed an end-to-end speech recognition system for Bangla language. We trained a long short-term memory (LSTM) network with an open sourced Bangla dataset provided by Google [22]. Given a sequence of audio signals, assuming the audio will be Bangla phonemes; the network will predict a sequence of the probability distribution of Bangla characters. The process is called language transcription which is regarded as one of the most difficult tasks of ASR.

1.2 Motivation

Bangla is one of the most spoken languages of the world, still being counted among the languages which are technologically under-resourced. This experiment, however, is an effort towards making Bangla language digitized. Being implemented correctly, the system will reduce a lot of backlog in courts and government offices by automatically transcribe the words spoken. In customer care and in call centers it can be implemented to handle customer queries automatically. Moreover, efforts of health workers in collecting data regarding vaccination and birth control can be minimized through Bangla language transcription. Not only that, people can integrate this system into their robots for understanding commands in Bengali. Also, by integrating this into any mobile application will allow the Bengali community people to communicate with that application easily and efficiently. However, for making a Bengali artificial chatbot Bengali transcription system can be very useful, which can be integrated later into various e-commerce website to improve the customer experience.

1.3 Thesis Outline

The entire senior design project is described within seven chapters. Each section covered about a certain task of the project which is summarized below,

(I) Chapter one gives an introduction of automatic speech recognition and transcription system as a whole, history and necessity of ASR or transcription system, the

engineering problem, and our main objective, finally our motivation behind doing this project.

(II) Chapter two covers an extensive literature review of previous works on both automatic speech recognition and transcription system, along with brief theoretical overview of Spectrogram, Mel-Frequency Cepstral Coefficients (MFCCs), Principle Component Analysis (PCA), Artificial Neural Networks (ANN), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM), and Connectionist Temporal Classification (CTC).

(III) Chapter three is about the technical description of the entire project. It includes detail explanation about the dataset, preprocessing and different model architectures for both Bengali ASR and transcription system.

(IV) Chapter four is about the implementation of our project. It provides comprehensive details about the training infrastructure, programming language, and libraries used for building both systems.

(V) Chapter five is regarding the results and analysis of both systems. It gives clear details of testing the project design and shows the output results achieved after implementing different model architecture for both Bengali ASR and transcription system.

(VI) Chapter six provide the societal and environmental impact of our project, means how our project will be affected or benefited the society.

(VII) Chapter seven is on discussion and future work. It includes the limitations and issues we faced to implement this entire project. It also describes possible future development of this project work.

CHAPTER 2

RELATED WORKS

Exploring the Neural network started more than two decades earlier. But, recently the use of deep learning has increased due to the advancement of the computing system. This work is inspired by previous work in this particular field including deep learning, speech recognition and transcription in other languages.

2.1 Spectrogram

Spectrogram is the most conventional technique to represent the spectrum of frequencies of a sound visually. A spectrogram shows how the frequency component of a sound signal changes over time. It is used extensively in speech processing and other fields where analyzing the audio or sound signal is an important task. However, to obtain the spectrogram from an audio signal, several steps need to follow. At first, an audio signal goes through a pre-emphasis filter to amplify the high frequencies which are very helpful for avoiding numerical problems during the Fourier transform operation. Then the signal gets sliced into short-time frames and a window function such as the Hamming window is applied to each frame. Afterward, the Fourier transform or more specifically Short-Time Fourier-Transform (STFT) has been applied to each frame to calculate the frequency spectrum and then compute the power spectrum (periodogram). The final step is to compute the filter banks on the power spectrum to extract the frequency bands. After doing all of these steps, a precise spectrogram can be obtained.

2.2 Mel-Frequency Cepstral Coefficients (MFCCs)

Mel-frequency cepstral coefficients (MFCCs) is widely used feature selection techniques. It is a prominent technique for building an Automatic Speech Recognition (ASR) system. However, the MFCC feature selection process contains several steps to extract the features from an audio signal. At first, an audio signal splits into short timestamps, since an audio signal varies too much in long timestamps. Then for each frame, the periodogram estimation of the power spectrum has been calculated. After calculating the periodogram estimation, the mel-filter bank is applied and summed the filter energy. Additionally, the logarithm of all filter bank energies has been calculated to find out the Discrete Cosine Transform (DCT) of these log filter bank energies. At the last, the lower DCT coefficients of range 2-13 have been kept for an automatic speech recognition system, since keeping the higher DCT coefficients reduce the performance of an ASR.

2.3 Principle Component Analysis (PCA)

Principle Component Analysis (PCA) is a classical and widely used technique to reduce the dimension of a feature set. PCA rotationally transform features of a dataset into a lower dimensional set which are not correlated with each other. These uncorrelated features are called principal components (PCs). In PCA, it is assumed that some of the variables in the dataset are linearly correlated, for which it tries to find orthogonal projects of the dataset. And then, it tries to find a lower dimensional surface which has the minimum projection error or the sum of squares onto that surface is minimized. However, PCA is extensively used to reduce the dimension of a dataset because the overfitting problem might occur due to the high dimension in the feature set. Additionally, too many dimensions are computationally expensive and extravagant for learning algorithms.

2.4 Artificial Neural Networks (ANN)

An artificial neural network composed of artificial neurons or nodes which is loosely inspired by the biological neural networks. A neural network is a framework for the different learning algorithm to work altogether. It can process complex input data and learn to execute the specific task without explicitly programmed. The basic structure of a neural network consists of several layers such as an input layer, an output layer, and several hidden layers. However, recently deep neural network got popular among

researcher for building an automatic speech recognition (ASR) system. Several work has been done on which some form of deep neural network is used [13, 30]. Additionally, there are several other types of neural network approaches that are being used in ASR among them Convolutional Neural Network shows some promising result [37, 39]. But, among all neural networks, Recurrent Neural Networks (RNN), particularly a variant of RNN called long short-term memory (LSTM) network have achieved extraordinary results in ASR, since acoustic model need to deal with (time) sequential feature inputs.

2.5 Convolutional Neural Networks (CNN)

The convolutional neural networks (CNN) are a simple extension of the multi-layer perceptron model which can be considered as a diverse version of the standard neural networks [7, 35]. In this section, we briefly discuss the architecture and other dimensions of traditional CNNs which are used mainly for speech recognition purposes.

2.5.1 CNN Layered Architecture

A typical CNN for automatic speech recognition (ASR) introduces a different kind of network infrastructure compared to other artificial neural networks (ANN) and deep neural networks (DNN). However, traditional CNN consists of layers stacked together which are an input layer, a group of convolutional and pooling layers, several fully connected layers, and finally an output layer [35]. The convolutional and pooling layers, followed by fully connected layers are the main differences of CNN compared to other neural networks, and this kind of special layer architecture has significant practical consequences in terms of speech recognition.

2.5.2 Convolution Layer

A convolutional layer organizes hidden layer such a way that can take the advantages of input layer which is in form of a two-dimensional input data. Each hidden unit of convolutional layer processes only a small part of the whole input space rather than connecting to all the inputs coming from the previous input layer and it applies some arbitrary filters of an arbitrary dimension which result in feature map. From the feature map, CNN can understand local features of the data.

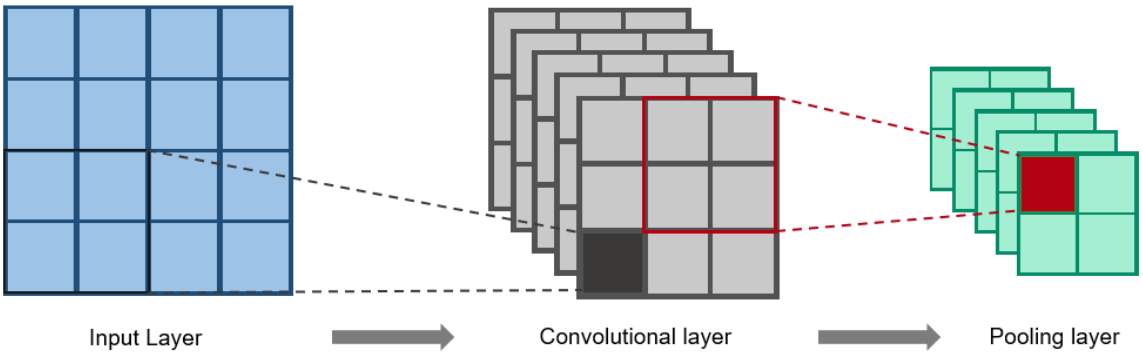

FIGURE 2.1. Convolutional Neural Network

2.5.3 Pooling Layer

Another layer is connected with one convolutional layer which is called pooling layer. Pooling layer reduces the dimensionality of the extracted feature maps by applying a window of an arbitrary size which is called stride. It can extract either max, average or the sum of the windows. In this case, max pooling is used which extracted the highest values for each window in the feature map.

2.5.4 Activation using rectified linear unit (ReLU)

Previously, logistic sigmoid and hyperbolic tangent have been used widely as non-linear activation functions in deep neural architectures like CNNs. But, recently some alternative solutions have emerged and the application of Rectified Linear Units (ReLUs) is one of the most commonly used alternatives. Additionally, ReLU is the common alternative solution, since it has several advantages over typical activation functions which are faster computation and more efficient gradient propagation, biological plausibility and sparse activation structure [15].

2.6 Recurrent Neural Networks (RNN)

In general neural network approach, all the feature input is considered independent. But, considering independent feature is not compatible for working with a speech to text

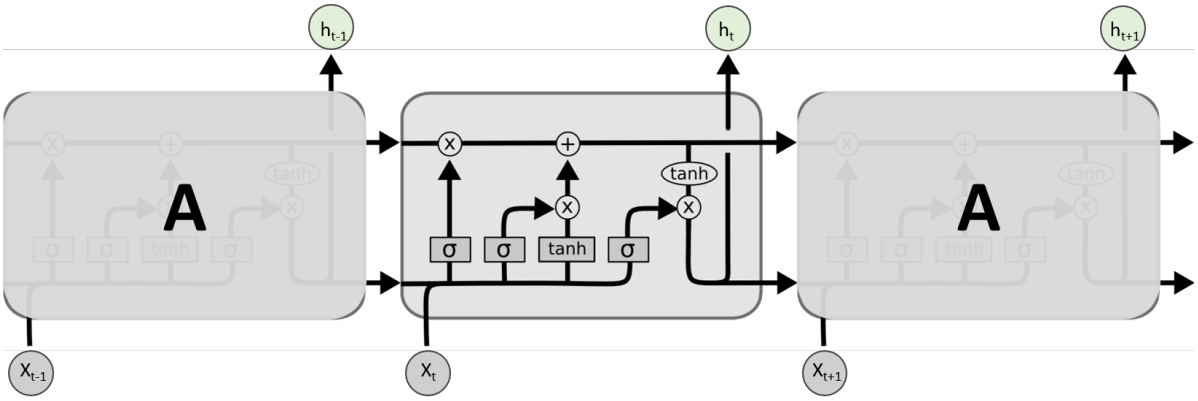

FIGURE 2.2. The repeating module in an LSTM. Figure reproduced from [32]

system where a model has to track sequential feature inputs in order to keep track of which input comes after what. Regarding this issue, a Recurrent Neural Network comes forwards, as RNN works better for keeping track of time sequential data. In other words, an RNN model keeps a designated space for some specific information and by using that information it keeps track of the next outcome. Though general RNN architecture can work with sequential input, there are some flaws to keep track of long-term dependencies [38]. Moreover, RNN can have both unidirectional and bidirectional recurrences.

2.6.1 Long Short-Term Memory (LSTM)

To overcome the limitations of a general RNN architecture, long short-term memory (LSTM) networks comes in front line which is a special variant of RNN that is capable of leaning long-term dependencies [16]. As LSTM can handle long-term dependencies, it is now a widely used architecture for solving such kind of problems which consist of sequential inputs like automatic speech recognition. An LSTM layer contains memory blocks those are recurrently connected with each other like figure 2.2.

2.7 Connectionist Temporal Classification (CTC)

Several types of loss function exist which are used in general neural network approaches. As speech recognition system or speech to text transcription deals with variable length audio inputs, Connectionist Temporal Classification (CTC) loss function coupled with an

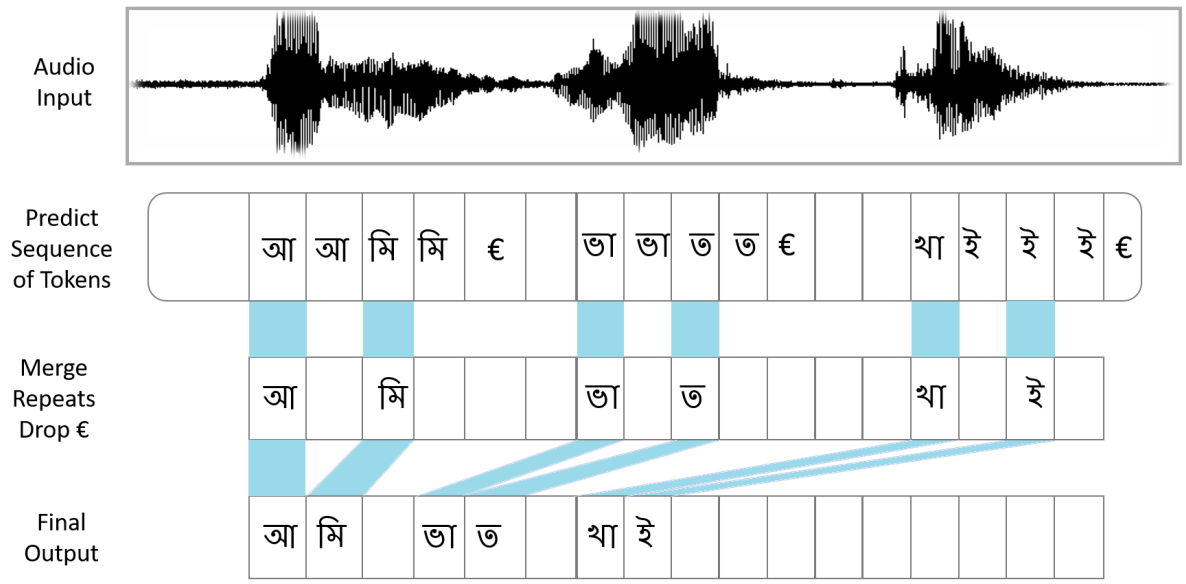

FIGURE 2.3. How CTC collapsing works. Figure reproduced from [17]

RNN or LSTM network is widely used for mapping the corresponding variable length audio input with the variable length output.

2.8 Theoretical Overview of Language Model

The language model learns the joint probability of the sequences of the language [10]. This paper [12] discusses n-gram models based on classes of words. It also explores statistical algorithms for assigning words to classes based on their co-existence with other words. In this paper [24] researchers presents a m-gram language model which offers a space and computation efficient solution of estimiting probabilities sparse data. A new recurrent neural network based language model has been presented here [27]. They found that by combining several recurrent language models they can obtain around 50 percent reduction on the perplexity. Moreover, they have encountered 18 percent reduction on the word error rate on a dataset they have experimented with. By extending the recurrent language model, in this paper [28] they have come to a modified model which is faster both during training and testing, smaller and more accurate than the earlier one. They have incorporated a back propagation through time (BPTT) algorithm and also have showed an empirical comparison with the feed forward neural network. Researchers in this paper proposed a fast fast hierarchical language model along with a

simple feature-based algorithm for automatic construction of word trees from the data. In [29], they have shown that this model can achieve state-of-the-art performance by outperforming all other hierarchical models.

2.9 Related Work on Automatic Speech Recognition

Due to the progression of computational powers, Neural Networks have been applied to many studies which achieved the extraordinary result for handling a large feature input. Among many different approaches, convolutional neural networks have been used previously for speech recognition tasks. CNN's reduce error rate by 6-10 percent compared to deep neural networks (DNN) which are found on this study [7] applied to the TIMIT phone recognition dataset. Moreover, they performed some experiments using full weight sharing (FWS) and limited weight sharing (LWS) schemes and it suggests that LWS is more effective as it can learn feature patterns of different frequency bands. In this paper [36] a very deep convolutional neural network has been applied for noisy speech recognition. They experimented with the sizes of filters, input feature map, and pooling layers to find the optimum setup. They evaluated the proposed model in Aurora4task and AMI meeting transcription dataset and found out that very deep CNN's reduce word error rate (WER) significantly for noisy speech recognition. However, in [14], they have explored something significantly interesting. They have trained a deep belief network (DBN) with unlabeled data of call routing task and used the learned features form this network to initialize a feed-forward neural network which fine-tunes itself by back-propagation. After that, they have compared three classic classifiers: support vector machine (SVM), maximum entropy (MaxEnt) and boosting with the DBN initialized network. They claim that the DBN initialized model has gained the accuracy which is equal to the best of the other baseline models. This study [23] outlines the possibility of using linear and log-linear stacking methods for ensemble learning for speech recognition using CNN, recurrent neural network (RNN) and fully connected DNN. [34] uses a DBN pre-trained neural network for large vocabulary speech recognition and claimed that it outperformed the baseline Gaussian mixture model-Hidden Markov model (GMM/HMM) which was built on a much larger dataset than the one they used. Many studies suggest that it is better to use a HMM model which can easily deal with the temporal differentiation of speech features. Studies also suggest that GMM models perform well with finding out each short time frame of audio input. But in this study [19] shows that using a DNN which consists of many hidden layers can easily outperform a

HMM-GMM based model by a large scale.

Although, when it comes to Bangla speech recognition, not much work has been done. There are, however, a few praiseworthy attempts. [31] proposed a model which calculates the linear predictive coding (LPC) and cepstral coefficients to form vector quantization which is then fed to an artificial neural network (ANN). They took audio samples of short speech from ten people in a noiseless condition. Their model has achieved a good performance recognizing known voices, but for unknown voices, the model's performance slightly degrades. In this study [20] they introduced a Bangladeshi accented Bangla digit automatic speech recognition system. They used Mel-frequency cepstral coefficients (MFCC) as a feature extraction method and feed the MFCC vectors to a HMM based classifier for recognition. However, in [35], they also built a digit recognizer and used MFCC analysis as a feature extractor but they feed those features to a back-propagation neural network instead. In this paper [8], they discussed four techniques of automatic speech recognition for Bangla words: MFCC, LPC, GMM and dynamic time wrapping (DTW). They compared these techniques in terms of recognition rate and elapsed time. MFCC feature extraction is used in most of the aspect as it outperforms other methods regarding short audio segment.

2.10 Related Work Regarding Transcription

Due to the advancement of computational power, there have been many progressive work done in the sector of speech recognition system for different speech corpus. We have seen various approaches applied to speech recognition, transcription and speech classification. Among these approaches some of those have shown satisfactory result. However, significant efforts has yet to be made for under resourced languages. This paper [42] proposes a two layered architecture for automatic speech recognition of Southeast Asian languages including Chinese, Thai and Vietnamese.The first layer has a dense phoneme network and the second layer does word decoding as the model extracts information on word level. A time delay neural network has been proposed for automatic speech recognition of Indonesian language in this paper [40]. On the other hand, in popular language e.g English, Mandarin etc, several progressive works has been done regarding transcription. Some of them are briefly described in the further sections below.

2.10.1 English

English being the international language, most of the work have done in this corpus. Recurrent Neural Network is the most used approach for end to end speech recognition. In [9] they have used RNN with multiple CNN layers, followed by several bidirectional recurrent layer and one fully connected layer before softmax layer. CTC loss function is used to handle a sequence of input. In [9, 18] CTC loss have been implemented for sequence prediction. Multiple types of feature extraction methods are used for these architectures. MFCC feature extraction works well with short speech commands [39]. But to work with a sequence of input data using spectrogram or log-spectrogram [9].

2.10.2 Mandarin

Mandarin is the most spoken language in the world. A speech recognition on this corpus is very complicated as there are a lot of alphabets to consider. Two different types of model is used in this paper [33] where Gaussian Mixture Model along with a Hidden Markov model and Deep neural network with a Hidden Markov Model is used. The result of DNN-HMM model surpasses GMM-HMM model. Moreover using log-spectrogram for feature extraction and with a recurrent model architecture [9] also produces promising result.

2.10.3 Hindi

Hindi language has some high lexical differentiation and also some ambiguous punctuation which differs in gender and number agreement. There are varieties phonetic pronunciation as well which creates complication while predicting a sequence of speech input. In order to work with a speech recognition system there are two models that must be considered. One is use of a acoustic model which deals with the acoustic features of the speech and another is the language model which deals with the sentence build up [25]. A different type of approach is proposed in this paper [25] where they mapped each Hindi character to the corresponding English pronunciation. MFCC feature extraction was used to train the system on both Hidden Markov model and Neural Network approach. A attention based model consist of an encoder decoder and an attention network was also used in this paper [41] which was able to predict speech without facing any major difficulties.

CHAPTER 3

TECHNICAL DESCRIPTION OF THE WORK

Deep learning techniques have been used to design the model for both Bangla short speech commands and Bangla transcription. For Bangla short speech commands, the model has been trained on the manually collected dataset. On the other hand for Bangla transcription, the model has been trained by curriculum training with an open source dataset.

3.1 Bangla Short Speech Commands

3.1.1 Dataset

The resources for Bangla speech recognition are not widely available while there has been previous work on Bangla speech recognition, the dataset employed are not available publicly for research purpose. therefore we collected a small datset. We choose the words that are used on a daily basis. There is a lot of work has been done using the English data set provided by Google which is known as the Speech Commands dataset. The dataset contains 65,000 samples. The duration of each short words is not more than 1 second. The dataset contains 30 words with a variation of over 1000 utterances of the public, who contributed through the AIY website. This dataset was used to pre-train our proposed model. However, we choose only 10 classes from the English data set. The short speech words in the mentioned dataset shown in Table 3.1.

We decided to choose words for our own dataset considering the similarity of the

Table 3.1: Short speech words in the mentioned data set

Bed	Go	On	No	One
Six	Three	Two	Yes	Zero

phonemes of each word that are available in the English data set. The reasoning behind choosing those particular words is to achieve a higher accuracy considering the model is already pre-trained on Speech commands data set and is admittedly a subjective procedure. We were not able to get a large number of data samples, as we collected the audio samples manually by going person to person. Our dataset contains 10 classes of data sample; the duration of each utterance is less than 2 seconds. We managed to get about 100 samples per class. The words we choose to train our model shown in Table 3.2.

Table 3.2: Short speech words in the mentioned data set

1	agerta	previous
2	aste	slowly
3	at	eight
4	baba	father
5	bame jao	go left
6	bari	house
7	basa	home
8	bon	sister
9	bondho koro	stop
10	boro	big

3.1.2 Model Architecture

We have experimented with several models and in all the models below, we have used categorical cross-entropy as loss function and adadelta as the optimizer.

3.1.2.1 MFCC Model

We have experimented with two kinds of feature extraction. We have extracted mel-frequency cepstral coefficients (MFCC) features from the audio files and feed the features to a convolutional neural network architecture; which we are considering to call the MFCC model. Before training our model with the MFCC inputs we normalized those inputs. The MFCC model has one convolutional layer with 5 filters and each of the filters

has a stride of 1. The convolutional layer is associated with a softmax layer which is the last layer of our model architecture. Regularization techniques like dropout of 0.25 and kernel regularizers l2 have been used to prevent the model from overfitting.

3.1.2.2 RAW Model

We have taken the raw audio files and feed them to a similar CNN architecture which we will call the raw model. The raw model has a similar architecture with one convolutional layer and a softmax layer. We applied a dropout rate of 0.8 in this model, which is the only difference between the MFCC model and the Raw model. The inputs of this model are the first 10000 values of each of the audio files since most audio files incorporate silences after 1 second. The inputs of this model, similar to the previous one, are being normalized as well.

3.1.2.3 Pre-trained Model

Our another approach was to use a pre-trained model before training with our data. Google has released a dataset of English short speech commands. We have extracted the MFCC features of the audio files and have trained a neural network with those features. The neural network has three convolutional layers which are associated with max-pooling and batch normalization layers. The first convolutional layer has 128 filters and second and third layers have 64 filters each. The filters of all three layers have a size of 3×3. We have saved the weights of the model while training and have used the weights as a mean for transfer learning.

3.2 Bangla Transcription

3.2.1 Dataset

The data set used in the proposed system is an open source data set taken from Openslr [22]. The dataset contains 2,18,703 utterances of Bengali speech with their corresponding labels in a CSV file. Although the dataset contains a lot of speech samples, we had to face some complication while working with it. Such as, some of the audio labels were missing and some of the audio files were very noisy. To overcome those problems we had to select only those convenient data that could be used for our architecture.

3.2.2 Audio Preprocessing

A neural network is capable of taking raw audio signals as inputs. However, raw audio signals have inconsistent phases of noises and silences. On the other hand, speech processing methods like MFCCs and Spectograms do linear transformations and as a result of that some information gets lost in the way. Therefore, researchers have to make some trade-offs. Nevertheless, in most cases they choose sophisticated signals like MFCCs, Spectograms, filterbanks etc. In this study we, however, experimented with raw audio signals and with processed audio inputs such as MFCCs and Spectograms as well. We have designed three different types of neural networks for these three kinds of inputs.

3.2.3 Model Architecture

We have experimented with three different kinds of neural network models in our study. One of our models which is comprised of several time distributed layers has relatively high parameters than the other models.

3.2.3.1 Model 1

The MFCCs extracted from the audio signals been feed to a time distributed dense layer which has 100 neurons on it. This layer has been followed by two more time distributed layers with 100 neurons on each of as them. Moreover, a bi-directional LSTM layer has been associated with the time distributed layers which are followed by a softmax layer which has as many neurons as the number of characters.

However, clipped rectified linear unit (ReLU) has been applied as the activation unit in the intermediate layers. A special kind of loss function, CTC, which is able to deal with the variable length of sequences, has been integrated with the model. Adam has been used as an optimizer.

3.2.3.2 Model 2

In this model, we have experimented with a relatively small neural network architecture. The MFCCs been passed into a one dimensional convolutional layer which has ten kernels. Each of the kernels has the size of three. Moreover, weight and activity regularizers have been applied to tackle overfitting.

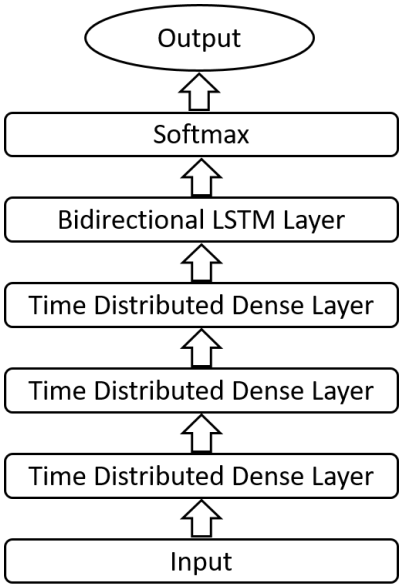

FIGURE 3.1. Block diagram of the model architecture.

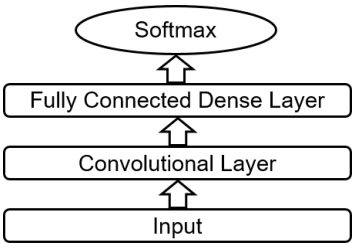

FIGURE 3.2. Block diagram of the CNN model architecture.

Rectified Lenear Unit(ReLU) has been used as a cost function and adam as the optimizer. CTC is used as a loss function here as well.

3.2.3.3 Model 3

In this architecture we have proposed a relatively bigger model in terms of layers than the other two models. The inputs has been passed through three one dimensional convolutional layers which have 50 neurons on each of them. The fetaures extracted by the convolutional layers then have been passed into six bi-directional LSTM layers. Finally, the featured are being processed by a time distributed dense layer which is associated with a CTC loss function.

The convolutional layers have used ReLU as the activation function whereas bi-directional

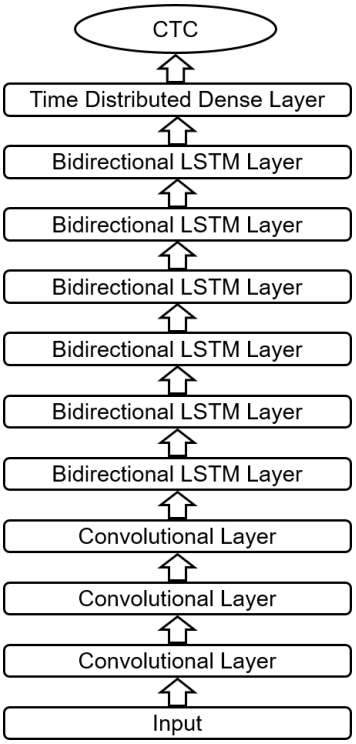

FIGURE 3.3. Block diagram of the model architecture.

LSTM layers have used clipped ReLU. Different regularization techniques like kernel and weight regularizers have been applied to the layers. Adam optimizer has been applied to this architecture.

3.2.4 Curriculum Training

Curriculum training method has been applied to train the model. Audios which have shorter sequences of characters get preference while being feed into the network. However, We started with audios which have as long as 10 characters on them and gradually increased the number of characters to be feed to the network. Curriculum training was proposed by researchers in this paper [11] by experimenting it in various setups. They have found that notable improvements in generalization can be achieved by implementing curriculum training. The underneath idea of this method of training is to start small by providing the model easier tasks to learn and then increase the difficulty level gradually.

Mathematically speaking, curriculum training can be viewed as a form of continuation

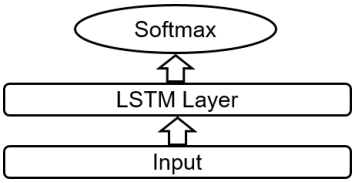

FIGURE 3.4. Block diagram of the language model architecture.

method. Continuation method can be derived as a strategy of optimizing non-convex functions globally. They hypothesize that this form of learning minimize the speed of convergence of the training process and if the trainable function is a non-convex one, the curriculum training has effects on the obtained global minima.

3.2.5 Language Model

We have used a language model to decode the outputs of the acoustic model in our pipeline. Also, we have designed a recurrent neural network architecture with bi-directional LSTMs beneath to implement the language model. The dataset released by Google which we have used to implement Bangla transcription engine is also used to train the language model. One hot encoding has been applied to process the text inputs of the dataset. We have removed some punctuation marks from the texts and ended up with seventy-eight unique characters. As the median length of Bangla words is four so we have designed a model which gives the probabilistic distribution of the fourth character given the first three letters of the word.

We have incorporated a recurrent neural network based architecture to implement the language model which has three bi-directional LSTM layers and one time distributed dense layer which has softmax as the activation function whereas the LSTM layers have clipped relu as their respective activation units. The loss function applied to this model is categorical cross entropy and adam has been used the optimizer.

CHAPTER 4

IMPLEMENTATION DETAILS

For implementing our proposed system, we try to build several architectures for both short speech commands recognition and transcription. Though we faced a lot of problem or issues, we try to build an efficient system where we used Convolutional Neural Network (CNN) for short speech command recognition and long short term memory (LSTM) has been used in transcription. All of these has been done in Python environment. A more elaborate description is provided in further sections below.

4.1 Short Speech Commands Recognition

We experimented with three kinds of approaches in recognizing short speech commands. In one of our approaches we have feed MFCC features to a CNN model and in another, we have not done any preprocessing rather fit the audios straight into the model. Moreover, we have leveraged transfer learning by loading a model which was initially trained with English short speech commands.

4.1.1 Programming Language and Libraries

We have used Keras [1] to design our models which is a high level neural network framework written on top of Python. It supports both convolutional and recurrent neural networks and their combination as well. Keras is capable of running on top of CNTK, Theano and Tensorflow. However, in our models we have used Tensorflow as the back-end.

We have used Librosa [2] for preprocessing of the audio files which is also written on top of Python. Librosa is capable of loading audios, extracting different kinds of mel features from the audios with a single line API call. It also can be used to extract and manipulate different spatial and temporal features of the audio files.

Moreover, Numpy [4] has been used for different scientific computing. Numpy is written on top of Python as well and has the capability of integrating C, C++ and Fortran code. Numpy is used mainly for linear algebra related calculations but it is capable of doing fourier transform and random number related calculations as well.

Tensorboard [6] is used a tool for visualizing the model's graphs and runs. Additionally, the decrease or increase of the loss and accuracy over time also can be visualized through tensorboard. However, tensorboard is a suite of web applications which comes along with the deep learning library Tensorflow.

Matplotlib [3] has been used to plot the the accuracies and losses. Matplotlib is a Python library capable of interactive 2D plotting of data.

4.1.2 Training Infrastructure

A graphical processing unit (GPU) of NVIDIA 1060 has been used to train the models. We have experimented with the optimizers, loss functions, batch size and learning rate a lot during the tuning time. In order to do that we had to load the audios and extract features from them every time we run the models which is a time exhausting process. To make the proceeding faster, we instead loaded and extracted MFCC features from the audios beforehand and save them as Numpy files. This has saved a lot of processing time as we just have to load the extracted features from the Numpy files to the models.

4.2 Bangla Transcription

Building a transcription system is a more complicated task than recognizing short speech commands. To build an efficient transcription system, we experimented with lots of different models where long short term memory (LSTM), a variant of Recurrent Neural Network (RNN) has been used. Everything has been done in Python environment. However, elaborate description of programming language, libraries, and training infrastructure are discussed in further sections below.

4.2.1 Programming Language and Libraries

In addition to the libraries mentioned in the short speech commands section we have used a Python library named Pandas [5] while designing the transcription pipeline. Pandas is an open source library for data structure and data analysis. Pandas is used to load and manipulate different kinds of structured data. In our pipeline, we have used Pandas to load the tab separated file (TSV) file where the name of the audio files and their annotations are stored.

4.2.2 Training Infrastructure

The hardware specification used for short speech commands recognition has also been used here. Additionally, we have used a free platform named Google Colaboratory [21] provided by Google to train some of our models. Google Colaboratory uses a Tesla K-80 GPU to train the models.

Similar to the short speech commands recognition we have also saved the MFCC and Spectogram features as Numpy files beforehand to fast track the training process.

Moreover, we have experimented by taking raw audios as input to the proposed neural network. Generally, librosa takes 1000 samples per second from the audio files. We have taken three different approaches while taking samples from raw audio files. They are given below,

(i) taking all the samples given by librosa as input.

(ii) taking every 10 samples from librosa as inputs.

(iii) taking every 100 samples from librosa as inputs.

CHAPTER 5

RESULTS AND ANALYSIS

Automatic speech recognition tasks have been evaluated by percentage accuracy, precision, recall, and F-1 score metrics. On the other hand, Transcription has been evaluated by Bilingual Evaluation Understudy Score (BLEU) and Character Error Rate (CER) respectively. Results of short speech command recognition task have been reported with elaborate analysis. However, results of transcription tasks will be reported after rigorous testing.

5.1 Bangla Short Speech Commands

We have trained the models using the training portion of the data and have tested them against the audio files which were being allocated for the testing purpose. Table 5.1 shows the performance of the models in terms of percentage accuracy. We see that the MFCC model performs better than the other models but the model suffers from over-fitting by a significant margin. The other two models achieve comparable test performance without any over-fitting.

However, we have run 1000 epochs over the whole data set to train the MFCC model. Figure 5.1 shows the epochs vs accuracy graph of the model. Moreover, we have plotted the loss generated in every epoch in figure 5.2.

The raw model is being trained by running 1000 epochs over the entire data set. The

Table 5.1: Performance of the models in percentage

Model	***Training Accuracy (%)***	***Testing Accuracy (%)***
MFCC	85.44	74.01
Raw	69.08	71.44
Transfer	68.06	73.00

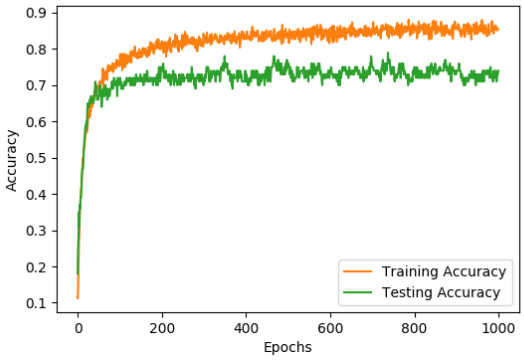

FIGURE 5.1. Epochs vs accuracy of MFCC model

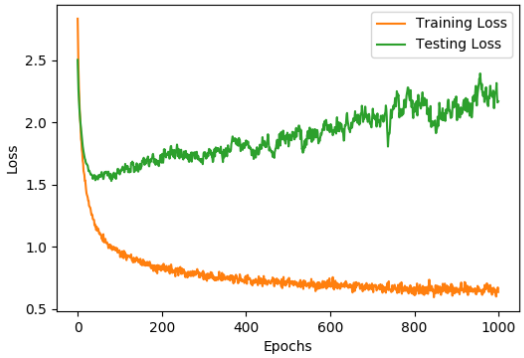

FIGURE 5.2. Epochs vs loss of MFCC model

accuracy and corresponding loss generated in every epoch are being plotted in Figure 5.3 and 5.4 respectively.

We loaded the weights generated while training the English dataset and retrained the model by running 1000 epochs over the Bangla short speech commands data set. Figure 5.5 and 5.6 represents the epochs vs accuracy and epochs vs loss graph of the transfer model respectively.

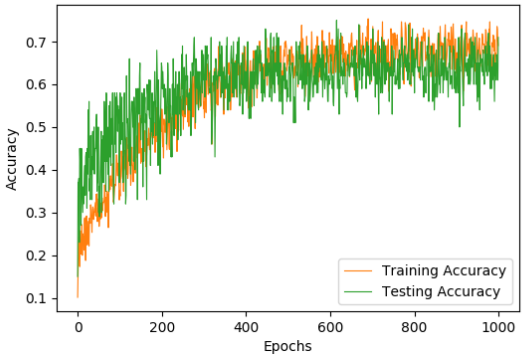

FIGURE 5.3. Epochs vs accuracy of RAW model

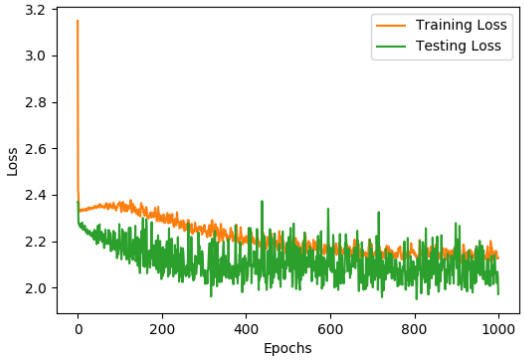

FIGURE 5.4. Epochs vs loss of RAW model

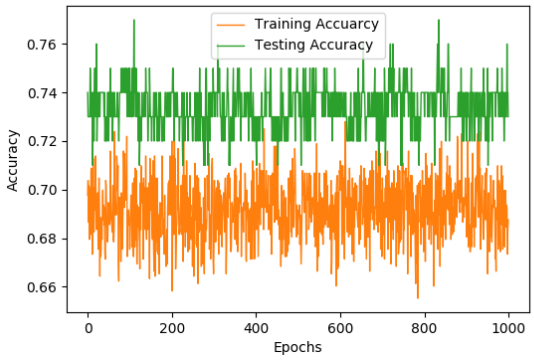

FIGURE 5.5. Epochs vs accuarcy of Transfer model

We have reserved 10 samples per class for testing purpose. We let the model predict the class label of these audio files and after that, we have generated confusion matrix

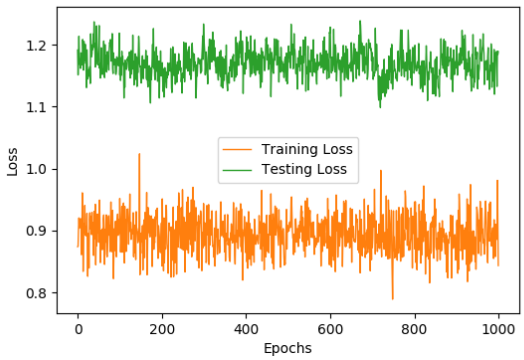

FIGURE 5.6. Epochs vs loss of Transfer model

from the predicted classes. Table 5.2 shows the confusion matrix of the MFCC model. We can see that the model does well in predicting "bari", "basa", "bon", and "boro" but failed miserably in recognizing "bame jao" and "bondho koro". The precision and recall per class of the model have been shown in table 5.3. Precision tells us about the classifier's ability not to label a negative sample as a positive one. Recall tells us whether the model is good at finding all the positive samples in the dataset.

Table 5.4 and 5.5 represents the confusion matrix and precision and recall metrics of

Table 5.2: Confusion Matrix of the MFCC model

	1	2	3	4	5	6	7	8	9	10
1	6	0	0	0	0	0	0	1	2	1
2	1	8	0	1	0	0	0	0	0	0
3	0	1	8	0	0	1	0	0	0	0
4	0	0	1	9	0	0	0	0	0	0
5	0	1	3	2	1	0	2	0	1	0
6	0	0	0	0	0	10	0	0	0	0
7	0	0	0	0	0	0	10	0	0	0
8	0	0	0	0	0	0	0	10	0	0
9	1	1	0	0	2	0	4	0	2	0
10	0	0	0	0	0	0	0	0	0	10

the raw model respectively. This model confuses between "agerta" and at but does well in recognizing "baba", "basa", "bon", and "boro". The model recognizes "bondho koro" as "bon" as their first syllable is the same.

Table 5.6 shows the confusion matrix of the transfer model and Table 5.7 represents

Table 5.3: Precison and recall of MFCC model

Class Label	***Precision***	***Recall***
1	0.75	0.60
2	0.73	0.80
3	0.67	0.80
4	0.75	0.90
5	0.33	0.10
6	0.91	1.00
7	0.62	1.00
8	0.91	1.00
9	0.40	0.20
10	0.91	1.00

Table 5.4: Confusion Matrix of raw model

	1	2	3	4	5	6	7	8	9	10
1	5	1	3	0	0	0	0	1	0	0
2	0	8	1	0	0	0	0	1	0	0
3	2	0	8	0	0	0	0	0	0	0
4	0	0	0	9	0	1	0	0	0	0
5	2	1	2	0	3	0	0	0	0	2
6	1	0	0	0	0	9	0	0	0	0
7	0	0	0	0	0	0	10	0	0	0
8	0	0	0	0	0	0	0	10	0	0
9	2	0	0	0	1	0	0	6	1	0
10	0	1	0	0	0	0	0	0	0	9

Table 5.5: Precison and recall of raw model

Class Label	***Precision***	***Recall***
1	0.42	0.50
2	0.73	.80
3	0.57	.80
4	0.75	.30
5	1.00	.90
6	0.75	.30
7	0.90	.90
8	1.00	1.00
9	0.56	1.00
10	0.82	.90

the precision and recall metrics. It also does well in recognizing "bari", "basa", "bon" but having difficulties identifying multi-syllable words "bame jao" and "bondho koro". The model has mistaken "agerta" as "bame jao" and "bari" as "bondho koro".

Table 5.6: Confusion Matrix of Transfer model

	1	2	3	4	5	6	7	8	9	10
1	9	0	0	0	0	1	0	0	0	0
2	1	8	0	1	0	0	0	0	0	0
3	2	0	7	1	0	0	0	0	0	0
4	0	0	0	9	0	1	0	0	0	0
5	3	1	1	1	1	0	2	0	0	1
6	0	0	0	0	0	10	0	0	0	0
7	0	0	0	0	0	0	10	0	0	0
8	0	0	0	0	0	0	0	10	0	0
9	0	0	0	1	0	3	1	0	2	3
10	1	0	0	0	0	1	0	0	0	8

Table 5.7: Precison and recall of Transfer model

Class Label	***Precision***	***Recall***
1	0.56	0.90
2	0.89	.80
3	0.88	.70
4	0.69	.90
5	1.00	.10
6	0.62	1.00
7	0.77	1.00
8	1.00	1.00
9	1.00	.20
10	0.67	.80

CHAPTER 6

SOCIETAL AND ENVIRONMENTAL IMPACT

In today's world, everything is getting automated. Automatic speech recognition and transcription systems are one of the best examples of the automated system in this present world. These systems already developed in many popular languages like English, Mandarin and etc. But in Bengali, the development of an automatic speech recognition system or the transcription system isn't yet as expected. Hence, an efficient speech recognition system, as well as a transcription system, can be very helpful in many fields. Also, it will be beneficial for the large Bengali community who speaks in Bengali. Its impact on society and the environment will be huge.

ASR or the transcription system get in the front line a lot earlier than today. And there are plenty of reasons for that. One reason is that the necessity of ASR or the transcription system among people increases over time. Additionally, it is a fact that the emergence of deep learning field helps to boost the development of these automatic systems. As like the English language, if an efficient ASR or transcription system can be built for the Bengali community, then it will be beneficial for many people and many fields as well. Some of the impacts given below.

(i) People can integrate the Bengali ASR in their robots so that it can understand command in Bengali as like as it understands the English command.

(ii) As like the English assistant, there could be possible to build a proficient Bengali assistant using the Bengali ASR and transcription system.

(iii) Bengali transcription system can be integrated on any mobile application which will help the Bengali community people to communicate with that application easily and efficiently.

(iv) Bengali transcription system also can be useful for making a Bengali artificial chatbot which can be integrated into various e-commerce website to improve the customer experience and it will boost their business as well.

(v) If it is possible to access or manipulate the mobile device using just Bengali command, then it will be beneficial for Blind and disable people along with normal people of the Bengali community and Bengali ASR or transcription system would be the first step towards this goal.

Therefore according to the above discussion, we can certainly understand, how an ASR and transcription system is important to our society for the present world. Because of the necessity of these systems and to keep moving forward with the present world, there must be a Bengali ASR and transcription system is needed for the Bengali community. And we hope that our proposed system will be the first little step towards this goal.

CHAPTER 7

DISCUSSION

In this final chapter, we briefly explain several complications & limitations related to our project and how we can make the current system more efficient by doing further developments in the future. After that, the concluding parts of the report is included.

7.1 Complications and Limitations

Deep learning provides us with some remarkable achievements. But, we faced several complications and difficulties to implement the whole project using deep learning. Although deep learning methodologies have achieved tremendous results, it does have several limitations. For example, a deep learning model can only estimate the assumption of a particular scenario but it can not be 100% accurate with predicting a particular class. A deep learning model faces many complications while learning with the limited example. And, it leads towards over-fitting. However, using deep learning is a little bit complicated in such problems that vary from time to time. Not only that, the computational power required to train each model is reasonably very high considering other approaches and it is a big issue for implementing our project. Because every-time we experimented with different models, it needed a lot of computational power to train the system. Additionally, it took a lot of time and space to train each system. However, deep learning algorithms perform remarkably well with labeled data, but it exaggerates against an unlabeled large amount of data.

Our system has few limitations as well. Such as, Bengali ASR system can't predict multi-syllable words. Currently, our transcription system doesn't transcript speech into text efficiently. Maybe, it needs a more sophisticated algorithm or model.

7.2 Future Work

In the future, we will work with more short speech data and improve our architecture to recognize multi-syllable words. We would like to build an efficient system which can do continuous audio transcription, means real time audio transcription. Also, we would like to build an Application Programming Interface, in short API so that everyone can access and get benefited from this system through that API. In different regions of our country, there is a lot of differentiation in Bengali pronunciation. Therefore, our final goal is to overcome these issues and build the transcription system for all regional verbal communication system.

7.3 Conclusion

Our report is based on the proposed system which is an end-to-end speech recognition system combined with a language model for Bengali community people. The whole system consists of two different major tasks, one of them is recognizing short speech Bengali commands, and the other one is to build a transcription system. So, our proposed approaches were to observe how different model architecture performs with relatively small or short speech data and a large dataset consist of sequential audio data. However, MFCC feature extraction performs well with predicting short speech command but it loses much information while extracting features from a sequence of audio input. On the other hand, Spectrogram feature extraction loses less information considerably and it can produce better results with sequential input features. In particular, three different approaches were explored for both of the data set. The first approach was to extract the MFCC features from the audio signals and used them to train the CNN model, whereas in the second approach just raw audio signals were used as the input to the model. The third approach attempted to leverage features learned from English speech using transfer learning. Among all models, the MFCC model had given slightly better test performance since it showed better percentage accuracy but suffered from the over-fitting issue. All the models had difficulties in recognizing multi-syllable words. Currently, we worked with the continuous audio data and we have built several model architectures

along with a language model on the Bengali language. However, our trained model could not achieve a good result with the sequential audio inputs. Perhaps, a more sophisticated model or approaches needed to work with sequential audio data. And, we hope that in near future we will be able to build an efficient transcription system along with an API which can be able to transcripted in real time so that Bengali people can get benefited from this system.

APPENDIX A

CODE

Listing A.1: Code snippets of MFCC

```
def wav2mfcc(file_path, max_len, n_mfcc):
    mfcc, sr = librosa.load(file_path, mono=True, sr=None)

    mfcc = librosa.feature.mfcc(mfcc, sr=16000, n_mfcc=n_mfcc)

    # If maximum length exceeds mfcc lengths then pad the remaining ones
    if (max_len > mfcc.shape[1]):
        pad_width = max_len - mfcc.shape[1]
        mfcc = np.pad(mfcc, pad_width=((0, 0), (0, pad_width)), mode='
            constant')

    # Else cutoff the remaining parts
    else:
        mfcc = mfcc[:, :max_len]

    return mfcc
```

Listing A.2: Model 3 architecture code snippets of Bangla transcription system

```
input_data = Input(name='the_input', shape=input_shape, dtype='float32')

```

```
conv = ZeroPadding1D(padding=(0, 2048))(input_data)
q = Conv1D(filters=32, name='conv_1', kernel_size=5, padding='valid',
    activation='relu', strides=1)(conv)
q = Conv1D(filters=32, name='conv_2', kernel_size=5, padding='valid',
    activation='relu', strides=1)(q)
q = Conv1D(filters=32, name='conv_3', kernel_size=5, padding='valid',
    activation='relu', strides=1)(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn1'), merge_mode='sum')(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn2'), merge_mode='sum')(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn3'), merge_mode='sum')(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn4'), merge_mode='sum')(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn5'), merge_mode='sum')(q)
q = Bidirectional(CuDNNLSTM(rnn_size, return_sequences=True,
    kernel_initializer='he_normal', name='birnn6'), merge_mode='sum')(q)

y_pred = TimeDistributed(Dense(output_dim, name="y_pred",
    kernel_initializer=init, bias_initializer=init, activation="softmax"),
    name="out")(q)

model1 = Model(inputs=input_data, outputs=y_pred)

labels = Input(name='the_labels', shape=[30], dtype='float32')
input_length = Input(name='input_length', shape=[1], dtype='int64')
label_length = Input(name='label_length', shape=[1], dtype='int64')

loss_out = Lambda(ctc_lambda_func, output_shape=(1,), name='ctc')([y_pred,
     labels, input_length, label_length])

model = Model(inputs=[input_data, labels, input_length, label_length],
    outputs=loss_out)
```

```
#model.load_weights('0_30_weights_big.h5')
model.compile(loss={'ctc': lambda y_true, y_pred: y_pred}, optimizer='adam
    ', metrics=['accuracy'])

outputs = {'ctc': np.zeros([len(x)])}

model.summary()
reduce_lr = keras.callbacks.ReduceLROnPlateau(monitor='loss', factor=0.2,
    patience=5, min_lr=0.0001)

model.fit([np.array(x), np.array(y_utf), np.array(input_length2), np.array
    (label_length2)], outputs,epochs=3500, batch_size=32,callbacks=[
    reduce_lr])
model.save_weights('0_30_weights_big.h5')
```

Listing A.3: Code snippets of the language model preprocessing

```
def data_onehot_x_y(annotation_arr, low, up):
  utf_to_onehot = onehot_mapping()
  data_x = []
  data_y = []

  for annotation in annotation_arr:
    if len(annotation) == 0:
      continue

    index = 0
    for char in annotation:
      utf = ord(char)

      x = []
      if index == 0:
        x.append(utf_to_onehot[-1])
        x.append(utf_to_onehot[-1])
        x.append(utf_to_onehot[-1])

```

```
20          data_x.append(x)
21          data_y.append(utf_to_onehot[utf])
22        elif index == 1:
23          x.append(utf_to_onehot[-1])
24          x.append(utf_to_onehot[-1])
25          utf_0 = ord(annotation[0])
26          x.append(utf_to_onehot[utf_0])
27
28          data_x.append(x)
29          data_y.append(utf_to_onehot[utf])
30        elif index == 2:
31          x.append(utf_to_onehot[-1])
32          utf_0 = ord(annotation[0])
33          utf_1 = ord(annotation[1])
34          x.append(utf_to_onehot[utf_0])
35          x.append(utf_to_onehot[utf_1])
36
37          data_x.append(x)
38          data_y.append(utf_to_onehot[utf])
39        else:
40          utf_1 = ord(annotation[index-3])
41          utf_2 = ord(annotation[index-2])
42          utf_3 = ord(annotation[index-1])
43          x.append(utf_to_onehot[utf_1])
44          x.append(utf_to_onehot[utf_2])
45          x.append(utf_to_onehot[utf_3])
46
47          data_x.append(x)
48          data_y.append(utf_to_onehot[utf])
49
50        index = index + 1
51
52  data_x = np.asarray(data_x)
53  data_y = np.asarray(data_y)
54
```

```
55  np.save("lan_model_onehot_"+str(low)+"_"+str(up)+"_x.npy", data_x)
56  np.save("lan_model_onehot_"+str(low)+"_"+str(up)+"_y.npy", data_y)
```

Listing A.4: Model architecture code snippets of the language model

```
1  def lan_model_lstm(X_train, X_test, y_train, y_test, low, up, weight):
2    model=Sequential();
3    model.add(CuDNNLSTM(100, return_sequences=True, input_shape=(3,147,)))
4    model.add(Flatten())
5    model.add(Dense(147, activation='softmax'))
6
7    if weight != None:
8      model.load_weights(weight)
9
10   model.compile(loss='categorical_crossentropy', optimizer = 'adam',
         metrics=['accuracy'])
11   model.summary()
12   model.fit(X_train,y_train, batch_size=500,epochs=200, verbose=1,
         validation_data=(X_test,y_test))
13
14   model.save_weights('lan_model_weight_'+str(low)+'_'+str(up)+'_v2.h5')
15
16   score,acc=model.evaluate(X_test, y_test, batch_size=500)
17   print("Accuracy", acc)
```

BIBLIOGRAPHY

[1] *Keras: The python deep learning library*.
Retrieved from https://keras.io/, Accessed: 2018-08-20.

[2] *Librosa*.
Retrieved from https://librosa.github.io/librosa/, Accessed: 2018-08-15.

[3] *Matplotlib*.
Retrieved from https://matplotlib.org/, Accessed: 2018-09-17.

[4] *Numpy*.
Retrieved from http://www.numpy.org/, Accessed: 2018-08-05.

[5] *Python data analysis library*.
Retrieved from https://pandas.pydata.org/, Accessed: 2018-10-15.

[6] *Tensorboard: Visualizing learning*.
Retrieved from https://www.tensorflow.org/guide/summaries_and_tensorboard, Accessed: 2018-09-30.

[7] O. ABDEL-HAMID, A. MOHAMED, H. JIANG, L. DENG, G. PENN, AND D. YU, *Convolutional neural networks for speech recognition*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 22 (2014), pp. 1533–1545.

[8] M. A. ALI, M. HOSSAIN, AND M. N. BHUIYAN, *Automatic speech recognition technique for bangla words*, International Journal of Advanced Science and Technology, 50 (2013).

[9] D. AMODEI, S. ANANTHANARAYANAN, R. ANUBHAI, J. BAI, AND ET. AL., *Deep speech 2 : End-to-end speech recognition in english and mandarin*, Proceedings of The 33rd International Conference on Machine Learning, 48 (2016), pp. 173–182.

[10] Y. BENGIO, R. DUCHARME, P. VINCENT, AND C. JAUVIN, *A neural probabilistic language model*, Journal of Machine Learning Research, 3 (2003), p. 1137–1155.

[11] Y. BENGIO, J. LOURADOUR, R. COLLOBERT, AND J. WESTON, *Curriculum learning*, Proceedings of the 26th Annual International Conference on Machine Learning, (2009), pp. 41–48.

[12] P. F. BROWN, P. V. DESOUZA, R. L. MERCER, V. J. D. PIETRA, AND J. C. LAI, *Class-based n-gram models of natural language*, Comput. Linguist., 18 (1992), pp. 467–479.

[13] G. DAHL, D. YU, L. DENG, AND A. ACERO, *Context-dependent pre-trained deep neural networks for large vocabulary speech recognition*, IEEE Transactions on Audio, Speech, and Language Processing, 20 (2012), pp. 30–42.

[14] L. DENG AND J. PLATT, *Ensemble deep learning for speech recognition*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, (2014).

[15] X. GLOROT, A. BORDES, AND Y. BENGIO, *Deep sparse rectifier neural networks*, Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 15 (2011), pp. 315–323.

[16] A. GRAVESA AND J. SCHMIDHUBER, *Framewise phoneme classification with bidirectional lstm and other neural network architectures*, International Joint Conference on Neural Networks (IJCNN), 18 (2005), pp. 602–610.

[17] A. HANNUN, *Sequence modeling with ctc*, nov 2017.
Retrieved from `https://distill.pub/2017/ctc/`, Accessed: 2018-12-08.

[18] A. HANNUN, C. CASE, J. CASPER, B. CATANZARO, G. DIAMOS, E. ELSEN, R. PRENGER, S. SATHEESH, S. SENGUPTA, A. COATES, AND A. Y. NG, *Deep speech: Scaling up end-to-end speech recognition*, CoRR, abs/1412.5567 (2014).

[19] G. HINTON, L. DENG, D. YU, G. E. DAHL, A. MOHAMED, N. JAITLY, A. SENIOR, V. VANHOUCKE, P. NGUYEN, T. N. SAINATH, AND B. KINGSBURY, *Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups*, IEEE Signal Processing Magazine, 29 (2012), pp. 82–97.

[20] M. A. HOSSAIN, M. M. RAHMAN, U. K. PRODHAN, AND M. F. KHAN, *Implementation of back-propagation neural network for isolated bangla speech recognition*, CoRR, abs/1308.3785 (2013).

[21] G. INC., *Google colaboratory*.
Retrieved from `https://colab.research.google.com/`, Accessed: 2018-10-20.

[22] G. INCORPORATED, *Large bengali asr training data set*, 2016.
Retrieved from `http://openslr.org/53`, Accessed: 2018-10-04.

[23] N. JAITLY, P. NGUYEN, A. SENIOR, AND V. VANHOUCKE, *Application of pre-trained deep neural networks to large vocabulary speech recognition*, 13th Annual Conference of the International Speech Communication Association, (2012), pp. 2578–2581.

[24] S. KATZ, *Estimation of probabilities from sparse data for the language model component of a speech recognizer*, IEEE Transactions on Acoustics, Speech, and Signal Processing, 35 (1987), pp. 400–401.

[25] M. KUMAR, N. RAJPUT, AND A. VERMA, *A large-vocabulary continuous speech recognition system for hindi*, IBM Journal of Research and Development, 48 (2004), pp. 703–715.

[26] K. F. LEE, *Automatic Speech Recognition: The Development of the SPHINX System*, Springer US, 1989.

[27] T. MIKOLOV, M. KARAFIÁT, L. BURGET, J. ČERNOCKÝ, AND S. KHUDANPUR, *Recurrent neural network based language model*, 11th Annual Conference of the International Speech Communication Association, (2010), pp. 1045–1048.

[28] T. MIKOLOV, S. KOMBRINK, L. BURGET, J. ČERNOCKÝ, AND S. KHUDANPUR, *Extensions of recurrent neural network language model*, 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (2011), pp. 5528–5531.

[29] A. MNIH AND G. E. HINTON, *A scalable hierarchical distributed language model*, Advances in Neural Information Processing Systems 21, (2009), pp. 1081–1088.

[30] A. R. MOHAMED, G. E. DAHL, AND G. HINTON, *Acoustic modeling using deep belief networks*, IEEE Transactions on Audio, Speech, and Language Processing, 20 (2012), pp. 14–22.

[31] G. MUHAMMAD, Y. A. ALOTAIBI, AND M. N. HUDA, *Automatic speech recognition for bangla digits*, 2009 12th International Conference on Computers and Information Technology, (2009), pp. 379–383.

[32] C. OLAH, *Understanding lstm networks*, Aug. 2015.
Retrieved from http://colah.github.io/posts/2015-08-Understanding-LSTMs/, Accessed: 2018-12-07.

[33] J. PAN, C. LIU, Z. WANG, Y. HU, AND H. JIANG, *Investigation of deep neural networks (dnn) for large vocabulary continuous speech recognition: Why dnn surpasses gmms in acoustic modeling*, 2012 8th International Symposium on Chinese Spoken Language Processing, (2012), pp. 301–305.

[34] A. K. PAUL, D. DAS, AND M. M. KAMAL, *Bangla speech recognition system using lpc and ann*, 2009 Seventh International Conference on Advances in Pattern Recognition, (2009), pp. 171–174.

[35] K. J. PICZAK, *Environmental sound classification with convolutional neural networks*, 2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP), (2015), pp. 1–6.

[36] Y. QIAN, M. BI, T. TAN, AND K. YU, *Very deep convolutional neural networks for noise robust speech recognition*, IEEE/ACM Transactions on Audio, Speech, and Language Processing, 24 (2016), pp. 2263–2276.

[37] T. N. SAINATH, A. MOHAMED, B. KINGSBURY, AND B. RAMABHADRAN, *Deep convolutional neural networks for lvcsr*, IEEE International Conference on Acoustics, Speech and Signal Processing, (2013), pp. 8614–8618.

[38] M. SCHUSTER AND K. K. PALIWAL, *Bidirectional recurrent neural networks*, IEEE Transactions on Signal Processing, 45 (1997), pp. 2673–2681.

[39] S. A. SUMON, J. CHOWDHURY, S. DEBNATH, N. MOHAMMED, AND S. MOMEN, *Bangla short speech commands recognition using convolutional neural networks*, International Conference on Bangla Speech and Language Processing (ICBSLP), (2018), pp. 1–6.

[40] M. TKACHENKO, A. YAMSHININ, N. LYUBIMOV, M. KOTOV, AND M. NASTASENKO, *Language identification using time delay neural network d-vector on short utterances*, Speech and Computer, (2016), pp. 443–449.

[41] S. TOSHNIWAL, T. N. SAINATH, R. J. WEISS, B. LI, P. MORENO, E. WEINSTEIN, AND K. RAO, *Multilingual speech recognition with a single end-to-end model*,

2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), (2018), pp. 4904–4908.

[42] Q. VU, K. DEMUYNCK, AND D. V. COMPERNOLLE, *Vietnamese automatic speech recognition: The flavor approach*, Chinese Spoken Language Processing, (2006), pp. 464–474.