Project/Thesis No:

CSE 4000: Thesis/Project

AN ADAPTIVE BENGALI VOICE CONTROLLED SYSTEM FOR AUTONOMOUS VEHICLE NAVIGATION

By

Naimur Rahman

Roll: 1907031

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh
October, 2024

An Adaptive Bengali Voice Controlled System for Autonomous Vehicle Navigation

By

Naimur Rahman
Roll: 1907031

A thesis submitted in partial fulfillment of the requirements for the degree of
"Bachelor of Science in Computer Science & Engineering"

Supervisor:

Dr. Mohammad Sheikh Sadi
Professor
Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna, Bangladesh.

_____
Signature

Department of Computer Science and Engineering
Khulna University of Engineering & Technology
Khulna 9203, Bangladesh
October, 2024

Acknowledgement

First of all, I would like to express my gratitude to the Almighty Allah for endowing me with the skills and capabilities necessary for undertaking and working on this thesis. I am thankful for the blessings that have enabled me to contribute to the field of this thesis.

I extend my deepest appreciation to Dr. Md. Sheikh Sadi, Professor of the Department of Computer Science and Engineering and the dedicated supervisor of this thesis. His profound expertise and steadfast guidance have been the bedrock of my journey in developing the thesis. His continuous support, invaluable insights, and encouraging feedback have played a pivotal role in advancing this thesis to its current state. His unwavering commitment to excellence, combined with his constructive criticism, has significantly shaped the trajectory of this thesis.

I am truly grateful for the enduring support and scholarly mentorship provided by Dr. Md. Sheikh Sadi sir. Without his enthusiastic motivation and steadfast encouragement, the successful projection of this thesis would not have been possible. His commitment to fostering a rich learning environment has been a source of inspiration throughout this thesis journey.

This thesis has been a collaborative effort, and I acknowledge the contributions of all those who have supported and inspired me throughout this endeavor.

Author

Abstract

This thesis explores the development of a Bengali voice command recognition system for offline real-time control of robotic vehicles, integrating concepts from machine learning and speech processing. The problem addressed is the lack of existing voice command systems tailored for Bengali, particularly those that can operate without an internet connection. To solve this, a CNN-LSTM hybrid model was designed and trained using Mel-Frequency Cepstral Coefficients (MFCC) extracted from voice data, augmented with techniques like pitch shifting and time-stretching. The model is capable of classifying five commands with an accuracy of 81%, and it demonstrates robustness to variations and defects in speech inputs, such as mispronunciations or noisy environments. The system's robustness, to noisy and defective inputs, demonstrates its potential for use in real-world, offline scenarios, such as robotic control and assistive devices.

Contents

List of Tables

List of Figures

CHAPTER I

Introduction

1.1 Problem Statement

The advancement of voice recognition technology has paved the way for more intuitive and hands-free control in various fields, including automotive systems. In regions like Bangladesh, where Bengali is the primary language, voice recognition systems tailored for the native language are underdeveloped, leading to limited accessibility. This thesis addresses this gap by developing a Bengali voice-controlled system for vehicular movement. The system aims to enable drivers to issue simple voice commands in Bengali for moving a vehicle forward, backward, left, right, and stopping. By leveraging modern machine learning tools and data augmentation techniques, this thesis demonstrates the potential for enhancing vehicle control and safety for Bengali-speaking users.

1.2 Background

Despite Bengali being spoken by millions, there has been minimal development in voice recognition systems tailored specifically for Bengali commands, particularly for real-time applications like vehicular control. Research in Bengali voice recognition has focused on general speech recognition or hybrid models integrating English and Bengali commands [1], [2]. Existing solutions predominantly cater to widely spoken languages or rely on cloud-based APIs, making them impractical in offline settings [3], [4]. Systems like CSVC-Net for code-switching [1] and IoT-based robotic vehicle control through Google Assistant [3] depend on internet connectivity, limiting their applicability in rural or resource-constrained environments. Virtual assistants such as Adrisya Sahayak [5] and Adheetee [6] focus on broader tasks for visually impaired users but are still cloud-dependent or general-purpose.

To address this gap, this thesis aims to build a lightweight, offline voice command system that can handle specific Bengali commands without relying on cloud services, ensuring real-time applicability for tasks such as robotic or vehicular control.

1.3 Objectives

- To develop a Bengali voice control system for vehicular movement using deep learning techniques.
- To recognize and classify different specific commands.
- To apply data augmentation techniques to enhance dataset diversity and model generalization.
- To achieve high command recognition accuracy using a CNN-LSTM architecture.
- To evaluate the system's performance in real-time scenarios for practical application.

1.4 Scope

The scope of this thesis is to design and implement an offline Bengali voice command recognition system using advanced machine learning techniques such as CNN-LSTM. The thesis focuses on developing a lightweight model that can operate on low-resource devices, like Raspberry Pi, for real-time control of robotic vehicles. By leveraging data augmentation and MFCC feature extraction, the model aims to accurately classify Bengali commands in noisy environments, providing a practical solution for voice-controlled systems without requiring cloud-based services or internet connectivity.

1.5 Unfamiliarity of The Problem

The problem of recognizing Bengali voice commands for vehicular control remains largely unexplored in both academic research and commercial applications. Most existing voice recognition systems have been developed for globally dominant languages such as English, Chinese, and Spanish, with limited attention paid to regional languages like Bengali. While some efforts have been made to develop general speech recognition systems in Bengali, there is a notable gap in solutions specifically designed for vehicular command recognition in this language.

This thesis addresses this gap by presenting an original solution focused on a Bengali voice control system for vehicular movements. Unlike previous work, which largely revolves around broader speech recognition, this thesis hones in on a practical, real-time application

of voice commands in a vehicular context. The approach employed here, which includes the use of a Convolutional Neural Network (CNN) combined with Long Short-Term Memory (LSTM) layers, represents a novel methodology in this domain. Moreover, the specific preprocessing techniques, such as noise reduction, pitch shifting, time stretching, and data augmentation, are tailored to enhance the accuracy and robustness of the system, making this solution unique and innovative.

This work has not been derived directly from any pre-existing research, making it a pioneering effort in the field of Bengali voice-controlled vehicular systems. It bridges the gap between traditional speech recognition systems and practical, language-specific applications in the transportation sector.

1.6 Thesis Planning

The planning for this thesis was structured around clear phases, from data collection and preprocessing to model development, testing, and evaluation. Each phase was carefully outlined to ensure the thesis progressed systematically, with milestones established to monitor key activities. The thesis followed an iterative approach, where adjustments were made based on the outcomes of each stage, allowing for flexibility in addressing challenges and optimizing the system's performance. Regular reviews of progress ensured the timely completion of tasks, from dataset augmentation to model training and evaluation.

The following Fig. 1.1 Gantt chart was designed for a projection of the thesis at panning level. In the context of this thesis, the Gantt chart was an essential tool for visualizing and organizing the entire thesis timeline. It mapped out key tasks such as data collection, preprocessing, model development, and evaluation, showing the start and end dates of each phase. By using a Gantt chart, the thesis’s workflow was structured in a way that allowed for efficient tracking of progress, identification of task dependencies, and proper resource allocation. This ensured that the thesis adhered to its schedule and met deadlines efficiently, providing a clear roadmap for each milestone.

The use of a Gantt chart was crucial in maintaining the balance between multiple tasks, identifying potential delays, and ensuring the thesis stayed on track throughout each phase of development.

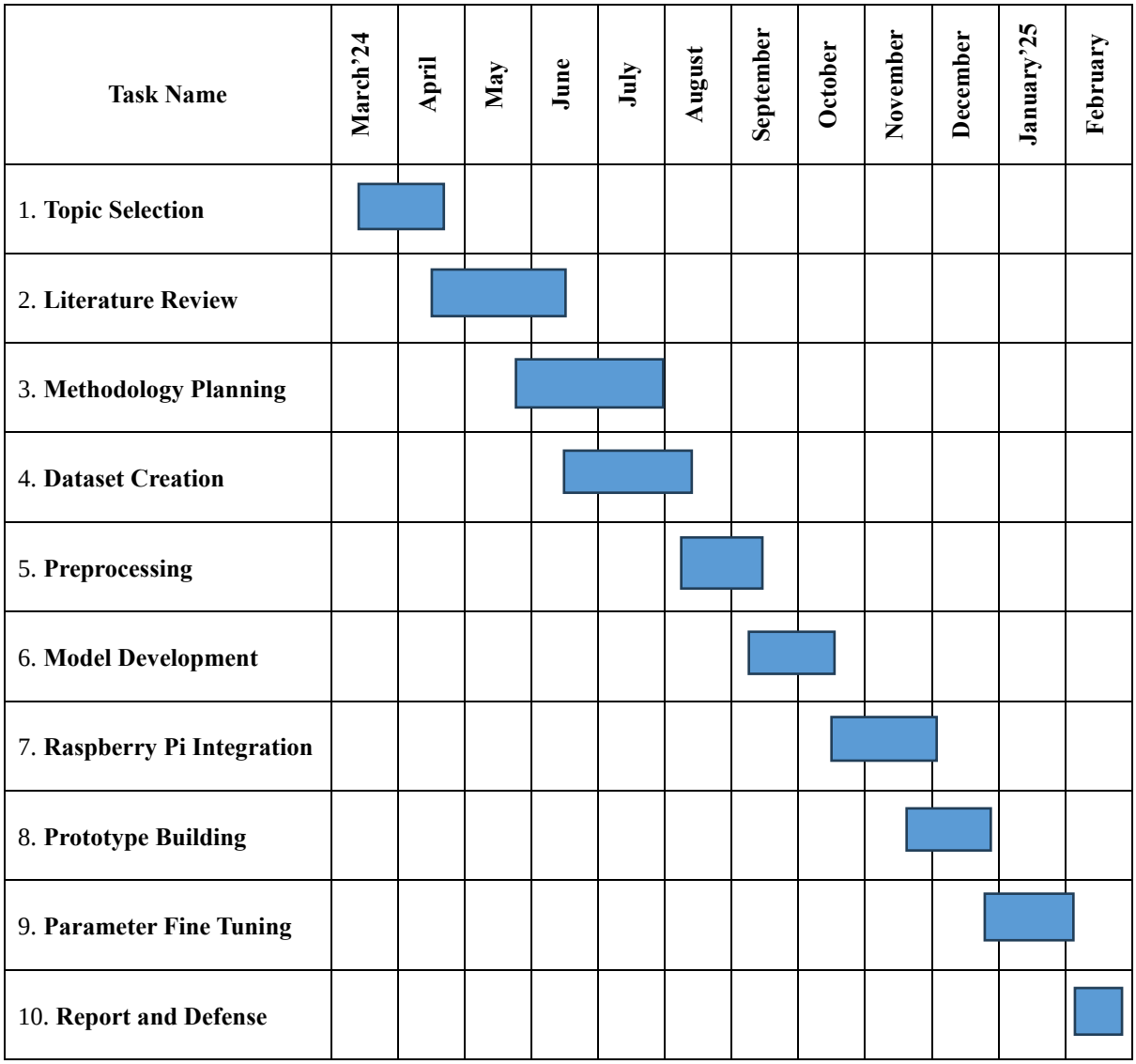


Fig. 1.1: Gantt chart for thesis planning.

The development of this Bengali voice-controlled system is particularly impactful when applied to small robotic vehicles, such as electric wheelchairs. By enabling voice-controlled navigation, the system enhances the mobility and independence of users, especially those with physical disabilities. The thesis adheres to strict guidelines to ensure societal, health, safety, legal, and cultural factors are considered throughout the design and implementation process.

1.6.1 Societal: This system aims to significantly improve the quality of life for individuals with mobility challenges, particularly within the Bengali-speaking community. Wheelchair users often struggle with manual control due to physical limitations, and this voice-controlled system offers an intuitive, hands-free alternative. By

incorporating voice commands, it provides a more accessible and user-friendly interface, making mobility devices easier to operate for a wider range of users, including the elderly and individuals with disabilities.

1.6.2 **Health and Safety**: Voice-activated control reduces the cognitive and physical effort required to operate a wheelchair. For users with limited hand or arm mobility, traditional joystick controls can be challenging. This voice-controlled system enhances safety by allowing users to focus on their surroundings, reducing the likelihood of accidents. The hands-free operation also helps to avoid distractions and enables the user to control the wheelchair even in situations where manual operation would be difficult or impossible.

1.6.3 **Legal**: In the context of small robotic vehicles like wheelchairs, the system complies with accessibility and assistive technology regulations. These technologies must meet specific standards for medical devices, ensuring they can be safely and effectively used by individuals with disabilities. The voice-controlled system is designed to adhere to these legal frameworks, making it a viable solution for integration into assistive mobility devices while ensuring compliance with relevant accessibility laws and guidelines.

1.6.4 **Cultural**: Focusing on Bengali voice commands ensures that this system is culturally inclusive and accessible to users in Bangladesh and other Bengali-speaking regions. Language barriers often limit the adoption of advanced technologies, particularly in healthcare and assistive devices. By providing a system tailored specifically for Bengali speakers, this thesis addresses a critical gap, ensuring that individuals who may not speak English can still benefit from advanced mobility solutions. This cultural relevance also helps to promote the broader adoption of the technology within the target population.

1.7 Organization of The Thesis

This thesis is organized into five chapters. Chapter 1 introduces the thesis, outlining the problem statement, objectives, and scope, with a focus on developing a Bengali voice control system for robotic vehicles. Chapter 2 reviews relevant literature, highlighting gaps in Bengali-specific voice recognition systems. Chapter 3 details the methodology, including

data collection, feature extraction, and model development using CNN-LSTM. Chapter 4 presents the results, analyzing model performance and accuracy. Finally, Chapter 5 concludes with key findings, limitations, and recommendations for future work, such as dataset expansion and hardware integration.

CHAPTER II

Literature Review

2.1 Introduction

Voice-controlled systems are becoming increasingly vital in assistive technology and small robotic vehicles. For Bengali-speaking users, especially those with physical limitations, voice-activated solutions offer significant benefits. However, despite advancements in speech recognition, the application of Bengali voice commands remains underexplored. This review synthesizes relevant research on voice command recognition systems and their applications in robotics, highlighting the gaps and positioning this study as a novel contribution to the field.

2.2 Existing Solutions

Yasmeen et al. [1] introduced CSVC-Net, a CNN-LSTM model designed to classify code-switched Bengali-English voice commands. The model relies on a dataset and focuses on processing bilingual commands. My model, built for Bengali-only commands and operates offline in a lightweight environment.

Berdibayeva et al. [2] explored speech command recognition using an artificial neural network (ANN). The study focused on feature extraction techniques like MFCCs and employed a BiLSTM architecture for speech recognition. It used the Google Speech Commands Dataset. My model differs by being tailored for Bengali-specific commands and optimized for low-resource environments.

Gupta et al. [3] developed an IoT-based system where a robotic vehicle is controlled through Google Assistant, making it dependent on voice APIs for executing commands via cloud computing. In contrast, my model is designed for Bengali commands and operates without external API dependence, ensuring complete offline functionality.

Chakraborty et al. [4] created a voice-controlled robotic car that uses voice commands through an Android app. The system focuses on Bluetooth connectivity and explicitly uses voice APIs for command recognition. My model is specifically built for Bengali commands,

offering offline functionality without relying on external services.

Sultan et al. [5] introduced Adrisya Sahayak, a virtual assistant designed for visually impaired Bengali speakers. While the assistant processes Bengali commands, it is implemented as a desktop application. My model emphasizes platform independence and suitability for resource-constrained environments.

Islam et al. [6] developed Adheetee, a Bangla virtual assistant for smartphones and personal computers. This assistant processes Bengali commands, but isn't adaptive or accurate enough. My model is tailored for Bengali commands, operating accurate enough to handle real-time command recognition.

Shawon et al. [7] designed a voice-controlled smart home automation system using Bluetooth technology. The system supports Bangla and English commands and relies on voice APIs for execution. My model differs by focusing specifically on Bengali commands and operating offline, independent of Bluetooth-based control modules.

Chowdhury et al. [8] evaluated Bangla speech recognition using methods like linear predictor coefficients and spectral analysis. This study focused on speech signal processing. My model advances this work by developing a Bengali command recognition system, optimized for offline, real-time applications.

Gawade et al. [9] proposed an in-vehicle speech command system for driver assistance using deep learning models to recognize and execute commands. The study does not rely on voice APIs but instead processes commands using deep learning techniques. My model, similarly, is Bengali-specific, ensuring robust performance in low-resource environments.

Sadeq et al. [10] presented an end-to-end system for recognizing Bangla voice commands using contextual rescoring. The system focuses on improving word error rates with deep learning models. My model similarly processes Bengali commands, but with a lightweight, offline-capable design.

Islam et al. [11] focused on detecting hate speech in Bengali using machine learning. This system, while processing Bengali language data, does not deal with voice commands on real-time. My model focuses on real-time Bengali command recognition and suitable for immediate voice command execution.

Khan et al. [12] developed a robotic car controlled via an Android app that processes human voice commands. The study relies on voice APIs, as it focuses on simple command execution using Android interfaces. My model, by contrast, is Bengali-specific, designed to operate offline and handle commands without requiring external services.

Azarang et al. [13] combined data augmentation methods, including speed perturbation and reverberation, to improve CNN-based voice command recognition. While it focuses on improving recognition performance, my model is Bengali-specific, for real-time execution.

Sumon et al. [14] focused on short Bangla speech commands using CNNs, with MFCC features for improved recognition. The study highlights how CNN models can effectively process Bengali commands. My model builds on this, offering offline functionality optimized for Bengali-specific commands.

Gupta et al. [15] developed a digital personal assistant for recognizing continuous Bangla voice commands using cross-correlation. The system focuses on processing commands using local computing resources. My model similarly focuses on Bengali commands, ensuring offline command execution in real-time.

Ramadan et al. [16] implemented an embedded system for detecting abusive Bengali speech using NLP and deep learning. Rahat et al. [17] developed a speech-controlled robot using Raspberry Pi, capable of recognizing multiple languages, including Bengali. The system explicitly relies on voice APIs. My model, built specifically for Bengali commands, works offline, making it ideal for low-resource, real-time control systems with good accuracy.

2.3 Discussion About Research Gaps

The analysis of all the research papers reveals that while many studies focus on speech recognition, none fully address Bengali-specific voice commands for offline operation, particularly in resource-constrained environments. Most existing solutions either focus on English/Bengali code-switching, use APIs for cloud-based processing, or handle general speech recognition without targeting Bengali command execution.

The novelty of this thesis lies in creating a lightweight, custom-built, offline-capable Bengali voice command system. This system fills the gap by being tailored specifically for Bengali commands, designed to operate without APIs, and capable of functioning entirely

offline, which is essential for real-time, low-resource applications like assistive technologies and small robotic systems. The Table 2.1 summarizes the gaps in each paper compared to the proposed model.

Table 2.1: Summarization of the gaps in each paper compared to the proposed model

Author & Ref. No.	Focus Area	Research Gap
Yasmeen et al. [1]	Bengali-English code-switched voice commands	More English focused rather than Bengali-only commands
Berdibayeva et al. [2]	Generic speech command recognition	No Bengali-specific commands; uses external datasets
Gupta et al. [3]	IoT-based vehicle control using Google Assistant	Dependent on cloud-based APIs; no Bengali-only focus
Chakraborty et al. [4]	Bluetooth-based voice control for robotic cars	No focus on Bengali; no offline functionality
Sultan et al. [5]	Bengali virtual assistant for visually impaired	Desktop-based; lacks offline functionality
Islam et al. [6]	Comprehensive Bangla virtual assistant	No offline command recognition
Shawon et al. [7]	Voice-controlled home automation using Bluetooth	Supports Bengali but uses Bluetooth-based APIs
Chowdhury et al. [8]	Speech recognition using spectral analysis	No command-specific system; focused on phoneme recognition
Gawade et al. [9]	In-vehicle driver assist system	English-focused; relies on cloud services
Sadeq et al. [10]	Bangla voice recognition using deep learning	No offline support; relies on deep learning models
Islam et al. [11]	Hate speech detection in Bengali	Focuses on classification, not command recognition

Khan et al. [12]	Human voice-controlled robotic car	No Bengali-specific commands; dependent on mobile APIs
Azarang et al. [13]	CNN-based voice command recognition	Uses generic data; no focus on Bengali commands
Sumon et al. [14]	Bangla short speech command recognition	No mention of offline functionality; bad accuracy
Gupta et al. [15]	Bangla personal assistant using cross-correlation	Desktop-based; uses more resources
Ramadan et al. [16]	Abusive speech detection in Bengali	Focuses on abusive speech detection, not command execution
Rahat et al. [17]	Speech-recognizable robot using Raspberry Pi	Uses cloud-based services; no Bengali-specific focus

This table illustrates the gaps in current research. Existing models either rely on APIs for processing or lack focus on Bengali-only commands. My proposed system is custom-built for Bengali-specific voice commands and is designed to work offline, making it a unique and necessary solution for real-time, low-resource applications like assistive technologies and small robotic vehicles.

CHAPTER III

Methodology

3.1 Introduction

The methodology section outlines the structured approach used to develop a voice command recognition system tailored for Bengali commands. This process includes the collection of a diverse dataset from multiple users, followed by comprehensive preprocessing steps such as noise reduction and data augmentation. The system employs Mel-frequency cepstral coefficients (MFCCs) for feature extraction and a CNN-LSTM model for classification. Each stage, from training to implementation, is optimized to ensure the model functions accurately and efficiently.

3.2 Dataset

The dataset used for this thesis was gathered from 50 different users, each contributing five voice recordings corresponding to the following commands: "Samne Jao" (move forward), "Pichone Jao" (move backward), "Dane Jao" (move right), "Bame Jao" (move left), and "Theme Jao" (stop). Each participant recorded one instance of each command, resulting in a total of 250 audio samples. The recordings were captured using participants' smartphones and then sent electronically for processing. This collection method ensured natural variations in speaker voice and background conditions, enhancing the dataset's diversity and robustness.

Participants were instructed to perform the recordings in various settings to introduce environmental diversity. Each audio sample was saved in WAV format and standardized for further processing. This approach ensured a robust dataset, capturing natural variations in speaker accents, tones, and environmental noise, which is essential for training the model to recognize Bengali commands in real-world applications. The diversity in speakers and conditions also contributed to the model's generalization across different voices and acoustic scenarios.

3.3 Preprocessing

Preprocessing and data augmentation are critical steps in preparing the audio data for the voice command recognition system. Preprocessing ensures the data is clean, consistent, and free from noise, which enhances the quality of features extracted from the audio. This includes noise reduction, normalization, and format standardization. Data augmentation artificially expands the dataset by applying transformations such as pitch shifting, time stretching, and adding noise. These techniques increase the model's ability to generalize to various real-world conditions, improving robustness and accuracy.

3.3.1 Noise Reduction Process

Noise reduction is a critical preprocessing step designed to isolate the main voice signal from background noise. For this thesis, an envelope-based noise reduction technique was applied, which works by identifying and suppressing low-amplitude, non-speech segments of the audio signal.

A. Raw Audio Signal

The input audio signal $y(t)$, sampled at a rate f_s of 16 kHz, contains both the desired speech and unwanted background noise. To reduce the noise, the signal undergoes several mathematical operations to filter out irrelevant parts and preserve the speech.

B. Absolute Signal Calculation

The first step in this process is computing the absolute value of the signal. The purpose of this operation is to transform all negative values (resulting from wave oscillations) into positive values, making it easier to determine the envelope of the signal that is given in 3.1.

$$y_{abs}(t) = | y(t) | \tag{3.1}$$

Where,

$y(t)$ is the original waveform,

$y_{abs}(t)$ is the absolute value of the signal.

C. Smoothing Using a Moving Average Filter

To estimate the overall energy of the signal and smooth out rapid oscillations, a moving average filter is applied. The moving average calculates the mean value of the signal across a short time window which is shown in 3.2. This filter helps to detect the broader trend in the signal while minimizing high-frequency variations that are usually noise. Mathematically:

$$E(t) = \frac{1}{N}\sum_{N}^{-N} y_{abs}(t+i) \qquad 3.2$$

Where,

N is the window size, typically set based on the sampling rate (e.g., 0.1 seconds),

$E(t)$ is the resulting smoothed envelope of the signal.

D. Thresholding

Once the envelope is computed, a threshold is applied to distinguish between the voice signal and the noise. This threshold T is chosen based on the average energy level of the audio. If the envelope of the signal at any point is below the threshold, that part of the signal is considered noise and is removed. The mask $M(t)$ is defined in 3.3 as:

$$M(t) = \begin{cases} 1 \; if \; E(t) > T \\ 0 \; if \; E(t) \leq T \end{cases} \qquad 3.3$$

Where,

$M(t)$ is a binary mask applied to the signal,

T is the threshold value, typically set empirically (e.g., 0.0005 for this thesis).

E. Applying The Mask

After determining which parts of the signal are noise (where $M(t) = 0$), the mask is applied to the original signal, keeping only the segments where the envelope exceeds the threshold demonstrated in 3.4.

$$y_{clean}(t) = M(t) \cdot y(t) \quad 3.4$$

This operation retains the dominant parts of the speech signal while reducing the background noise.

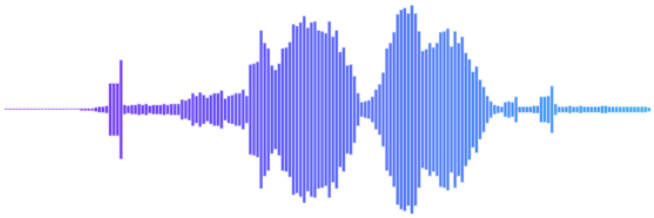
F. Theoretical Explanation of The Envelope

The envelope of a signal represents the outline of its amplitude variations. In speech signals, the envelope typically follows the intensity of spoken words and pauses, while the rapid oscillations in the waveform represent the high-frequency details of the sound (such as phonemes and tones). By calculating the envelope, we can identify parts of the signal with high energy (which typically correspond to speech) and distinguish them from parts with low energy (usually noise or silence).

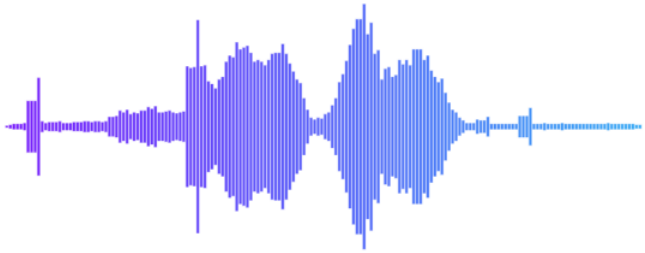
G. Mathematical Benefits of The Approach

The envelope-based noise reduction technique is computationally efficient because it works directly on the amplitude of the signal, avoiding the need for complex frequency-domain transformations (such as Fourier Transforms). It preserves the speech quality by only suppressing the low-energy background noise and is especially useful in situations where the noise level fluctuates across the recording.

The following Fig. 3.1 shows the differences before and after cleaning a raw audio signal where we can see that the less frequent noises are trimmed and the random shifting in a particular situation is averaged out and the clean signal is smoother and tolerable to random spiking in energy. In (a) the raw signal has low power level amplitudes that are basically considered as noise where as in (b) the lower powered signals are trimmed out and as well as it has a more consistent signal level compared to the raw signal.



(a)



(b)

Fig. 3.1: Raw signal after noise reduction process (a) raw signal and (b) cleaned signal

By reducing noise using this method, the system becomes more robust and accurate, especially in real-world scenarios where background noise can vary significantly.

3.3.2 Standardization of Audio Files

All audio files were converted to a consistent format to ensure compatibility during feature extraction. The following standard was applied:

- Sampling rate: 16 kHz (common for speech processing tasks)
- Bit depth: 16-bit PCM
- File format: WAV

The goal of this step is to maintain uniformity across the dataset, which is essential for ensuring that subsequent steps, such as feature extraction, are accurate and comparable across recordings.

3.4 Data Augmentation

Data augmentation plays a vital role in enhancing the diversity of the dataset by artificially creating variations of the original audio files. In this thesis, techniques such as pitch shifting, time stretching, and adding Gaussian noise were employed. These augmentations simulate different real-world conditions and speaking styles, which allows the model to generalize better and perform well under varied environments. Below is a detailed explanation of each augmentation technique, along with its mathematical basis.

In this thesis, data augmentation expanded the original dataset from 250 samples (50 samples per command) to 1,500 samples (300 per command). Each audio sample underwent three augmentation techniques: pitch shifting, time stretching, and adding Gaussian noise, increasing the effective size and diversity of the dataset.

3.4.1 Pitch Shifting

Pitch shifting was applied by shifting the audio pitch by +5 semitones and -5 semitones for each command. This resulted in an additional 100 samples per command (50 from upward and 50 from downward shifts).

Pitch shifting is performed in the frequency domain. First, a Fourier Transform (FT) is applied to the time-domain signal $y(t)$, converting it into its frequency-domain representation $Y(f)$. The pitch of the signal is then shifted by a factor $2^{n/12}$, where n represents the number of semitones to shift (positive for upward pitch shift, negative for downward) demonstrated in 3.5.

$$f' = f \cdot 2^{\frac{n}{12}} \tag{3.5}$$

Where,

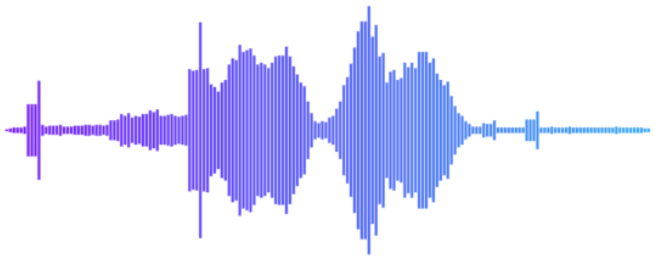
f is the original frequency,

f' is the new frequency after shifting,

n is the number of semitones (e.g., +5 semitones or -5 semitones for this study).

The transformed signal is then converted back into the time domain using the Inverse Fourier Transform (IFT). By altering the pitch, the model learns to recognize commands from speakers with varying vocal frequencies.

The Fig. 3.2 demonstrates the high and low pitch shifting applied to the cleaned signal with (a) signaling the clean signal, (b) indicating after low pitch shifting by -5 semitones and (c) indicating after high pitch shifting by +5 semitones.



(a)

(b)

(c)

Fig. 3.2: Signals after applying pitch shifting (a) cleaned signal, (b) after low pitch shifting and (c) after high pitch shifting

3.4.2 Time Stretching

Time stretching changed the speed of each audio sample without altering the pitch, simulating faster and slower speech. The stretching factors applied were 1.2x (faster) and 0.6x (slower), adding another 100 samples per command.

Time stretching involves modifying the rate of the signal's playback. The process can be represented by re-sampling the audio at a different rate. If the original signal is $y(t)$, time stretching with a factor α alters the time dimension the equation is demonstrated in 3.6.

$$y_{stretched}\,(t) = y\left(\frac{t}{\alpha}\right) \qquad 3.6$$

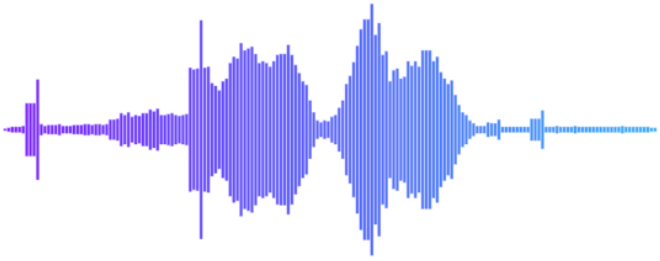
Where,

$\alpha > 1$ speeds up the signal (e.g., $\alpha = 1.2$ makes the audio faster),

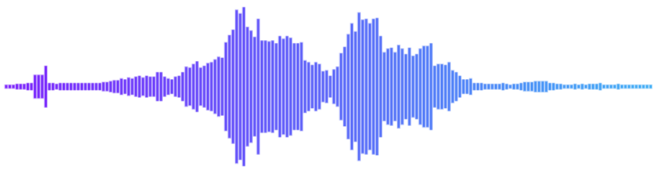
$\alpha < 1$ slows down the signal (e.g., $\alpha = 0.6$ makes the audio slower).

In this thesis, both faster (1.2x) and slower (0.6x) versions of each audio sample were created. Time stretching modifies the temporal dynamics of the audio without affecting the pitch, ensuring that the model can handle varying speech rates.

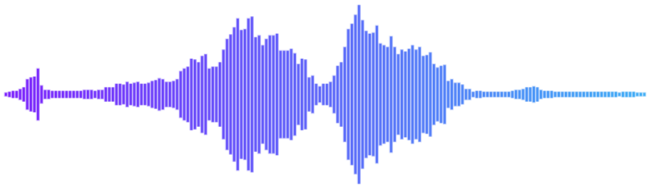
The Fig. 3.3 demonstrates the fast and slow time stretching applied to the cleaned signal with (a) signaling the clean signal, (b) indicating after fast time stretching by 1.2x and (c) indicating after slow time stretching by 0.6x.



(a)



(b)



(c)

Fig. 3.3: Signals after applying time stretching (a) cleaned signal, (b) after fast time stretching by 1.2x and (c) after slow time stretching by 0.6x

3.4.3 Adding Gaussian Noise

Gaussian noise was added to each sample, resulting in 50 additional samples per command.

This augmentation simulates noisy environments by adding random Gaussian noise to the original signal. This helps the model become more robust when recognizing commands in environments with background noise.

Gaussian noise is modeled as a random variable following a normal distribution in 3.7.

$$Noise \sim N(0, \sigma^2) \tag{3.7}$$

Where,

$\mu = 0$ is the mean (indicating zero-centered noise),

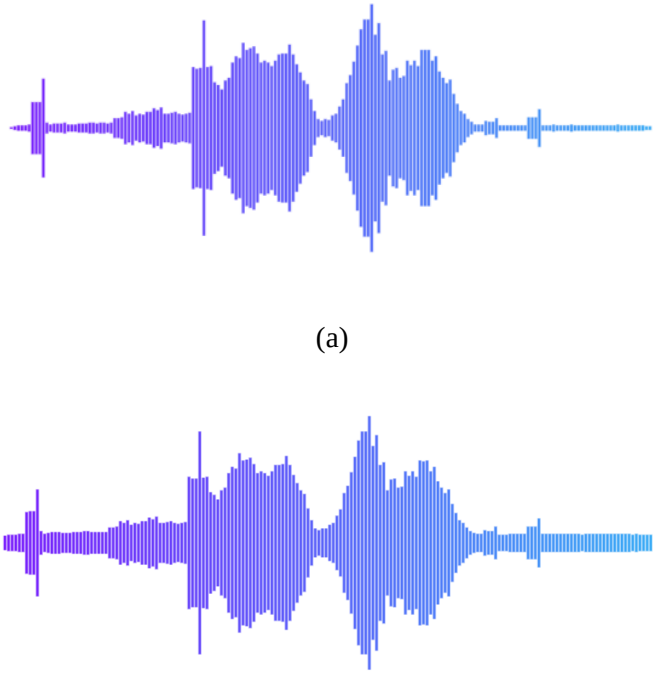
σ^2 is the variance, which controls the intensity of the noise.

The noisy signal $y_{noisy}(t)$ is computed by adding the Gaussian noise to the original signal as given in 3.8.

$$y_{noisy}(t) = y(t) + N(0, \sigma^2) \tag{3.8}$$

In this thesis, a small variance σ^2 (e.g., 0.001) was used to ensure the noise is subtle but present. This process mimics real-world conditions where background noise, such as environmental sounds or other speakers, might interfere with voice commands.

The Fig. 3.4 demonstrates the random noise applied to the cleaned signal with (a) signaling the clean signal, (b) indicating after the random noise is added to the signal.



(a)

(b)

Fig. 3.4: Signals after applying random noise (a) cleaned signal and (b) signal with random noise

3.4.4 Summary

By applying these augmentation techniques, the dataset's effective size is expanded beyond the original 250 samples. This augmentation increases the model's exposure to variations in tone, speed, and background noise, which are common in real-world scenarios. The mathematical basis for each augmentation ensures that key properties of the audio signal, such as the pitch or duration, are preserved while introducing controlled variations. These augmentations help improve the model's generalization, robustness, and accuracy when deployed in diverse environments.

Through these augmentation techniques, the original 250 samples (50 per command) were expanded to 1,500 samples (300 per command), providing the model with a diverse and

comprehensive dataset, improving its ability to generalize across different acoustic conditions and speaker variations.

3.5 Feature Extraction

In this thesis, Mel-Frequency Cepstral Coefficients (MFCCs) are the primary feature extraction technique, a method well-suited to speech recognition tasks. MFCCs transform the raw audio signal into a compact set of features that mimic the human auditory system. Here's a detailed breakdown of the process with mathematical formulations and theoretical explanations.

3.5.1 Framing and Windowing

Speech signals are non-stationary, meaning their characteristics change over time. Therefore, the audio signal is divided into short overlapping segments called frames to capture the short-term characteristics of the speech signal. Typically, a frame lasts 20 to 40 milliseconds, as speech characteristics are relatively stable over such short intervals.

Mathematically, if $y(t)$ is the original audio signal, it is split into frames $y_k(t)$ using a window function $w(t)$ that is shown in 3.9.

$$y_k(t) = y(t) \cdot w(t - kT) \tag{3.9}$$

Where,

k is the frame index,

T is the frame shift,

$w(t)$ is a window function, often a Hamming window shown in 3.10.

$$w(t) = 0.54 - 0.46 \cos\left(\frac{2\pi t}{N-1}\right) \tag{3.10}$$

This helps minimize spectral leakage by tapering the signal smoothly at the edges of each frame.

3.5.2 Fourier Transform (Short-Time Fourier Transform - STFT)

Each frame $y_k(t)$ is transformed into the frequency domain using the Discrete Fourier Transform (DFT) to compute the magnitude spectrum. This converts the time-domain signal into a representation that shows how much energy is present at different frequencies.

The DFT of a frame $y_k(t)$ is given in 3.11.

$$Y_k(f) = \sum_{n=0}^{N-1} y_k(n) e^{-\frac{j2\pi fn}{N}} \quad 3.11$$

Where,

$Y_k(f)$ is the Fourier coefficient at frequency f,

N is the number of points in the DFT (usually the frame length),

j is the imaginary unit,

$e^{-\frac{j2\pi fn}{N}}$ represents the Fourier basis.

The output is a complex number, but we are interested in the magnitude spectrum is given in 3.12.

$$| Y_k(f) | = \sqrt{Re\big(Y_k(f)\big)^2 + Im\big(Y_k(f)\big)^2} \quad 3.12$$

Where,

$Re(Y_k(f))$ and $Im(Y_k(f))$ are the real and imaginary components of $Y_k(f)$.

3.5.3 Mel Filter Bank

Human hearing perceives frequencies in a non-linear way, being more sensitive to lower frequencies and less sensitive to higher ones. The Mel scale mimics this characteristic of human auditory perception. The Mel scale is approximately linear below 1,000 Hz and logarithmic above 1,000 Hz.

The relationship between frequency in Hertz f and frequency in Mels f_{mel} is given in 3.13.

$$f_{mel} = 2595 \cdot log_{10}\left(1 + \frac{f}{700}\right) \quad 3.13$$

A Mel filter bank is applied to the magnitude spectrum. The filter bank consists of a series of triangular filters spaced along the Mel scale. The energy in each frequency band is computed by multiplying the magnitude spectrum by the filter bank and summing the result for each filter.

Mathematically, the output of each Mel filter $S_{mel}(m)$ is shown in 3.14.

$$S_{mel}(m) = \sum_{f=f_{min}}^{f_{max}} | Y_k(f) | H_m(f) \quad 3.14$$

Where,

$H_m(f)$ is the mth triangular filter, and f_{min} and f_{max} are the frequency bounds of the filter.

3.5.4 Logarithmic Compression

Humans perceive loudness on a logarithmic scale, so the energy in each Mel-filtered band is compressed using a logarithmic function. This step reduces the range of values and emphasizes lower energy components.

The logarithmic compression is mathematically expressed in 3.15:

$$S_{log}(m) = log\big(S_{mel}(m)\big) \quad 3.15$$

Where,

$S_{log}(m)$ is the log-scaled energy for the mth filter. This step ensures that loud sounds do not dominate the features, and smaller, subtler variations in speech are retained.

3.5.5 Discrete Cosine Transform (DCT)

To reduce the redundancy in the log-Mel features and produce a compact feature representation, the Discrete Cosine Transform (DCT) is applied. The DCT decorrelates the log-Mel features, and the result is the Mel-Frequency Cepstral Coefficients (MFCCs).

The DCT of the log-Mel features is given in 3.16.

$$C(n) = \sum_{m=0}^{M-1} S_{log}(m) \cdot \cos\left[\frac{\pi n}{M} \cdot (m + 0.5)\right] \quad 3.16$$

Where,

$C(n)$ is the nth MFCC,

M is the number of Mel filters.

The first few MFCCs (typically the first 12 or 13) capture the broad spectral features of the audio, which are critical for distinguishing between different voice commands.

3.5.6 Delta and Delta-Delta Coefficients

To capture temporal information about the rate of change in the speech signal, delta and delta-delta (acceleration) coefficients are computed. These coefficients help the model learn how the speech signal evolves over time, which is particularly useful for recognizing commands in varying speech styles.

The delta coefficient $\Delta C(n)$ is computed in 3.17:

$$\Delta C(n) = \frac{\sum_{i=1}^{T} i \cdot \left(C(n+i) - C(n-i)\right)}{2\sum_{i=1}^{T} i^2} \quad 3.17$$

Where,

T is the size of the window over which the difference is computed,

$C(n)$ are the static MFCCs.

The delta-delta coefficients $\Delta^2 C(n)$ are calculated in the same manner as the delta coefficients but applied to the delta values themselves.

3.5.7 Final Feature Vector

The final feature vector for each frame consists of the static MFCCs, delta coefficients, and delta-delta coefficients, resulting in a rich feature representation that captures both spectral and temporal dynamics that is shown in 3.18.

$$Feature\ vector = [C_1, C_2, \dots, C_{12}, \Delta C_1, \dots, \Delta C_{12}, \Delta^2 C_1, \dots, \Delta^2 C_{12}] \quad 3.18$$

These feature vectors serve as input to the machine learning model (CNN-LSTM in this case) for classification.

The use of MFCCs, combined with delta and delta-delta coefficients, ensures that the model captures both the spectral and temporal characteristics of Bengali voice commands. This feature extraction process reduces the complexity of the audio signal while retaining the key elements necessary for accurate voice command recognition.

3.6 Model Development

The model developed for recognizing Bengali voice commands uses a CNN-LSTM hybrid architecture. This architecture leverages the strengths of Convolutional Neural Networks (CNNs) for extracting local patterns from the input MFCCs and Long Short-Term Memory (LSTM) layers to capture the temporal dependencies in speech sequences. Below is a detailed explanation of the model layers and why they were chosen.

3.6.1 Batch Normalization

The first layer is a Batch Normalization layer applied to the input. The first batch normalization layer ensures that the input data is normalized across the batch. This improves the stability of the network by reducing the internal covariate shift and speeding up training. Batch normalization is also applied after LSTM layers to stabilize training, especially for deep models.

3.6.2 Convolutional Layers and Max Pooling

The model includes three Conv2D layers with increasing filter sizes (32, 64, and 128). The CNN layers focus on extracting local spatial features from the MFCC input. The MFCCs represent the spectral content of the speech signal over time, and the convolutional filters

(kernels) can learn to detect important local patterns, such as formants, which are characteristic frequencies in speech. These spatial patterns help the model distinguish between different voice commands.

By stacking multiple Conv2D layers (32, 64, 128 filters), the network is able to capture more complex and higher-level features at each successive layer. The network starts by identifying basic patterns like edges and builds toward recognizing more abstract features in the deeper layers.

Max Pooling is used after the convolutional layers to reduce the spatial dimensions of the feature maps. Pooling layers reduce the spatial dimensions of the feature maps. This reduction helps the model become more efficient by discarding unnecessary information, keeping only the most significant features, and reducing overfitting. MaxPooling also makes the network more invariant to small shifts or distortions in the input, such as slight variations in how the command is spoken.

3.6.3 Reshape Layer and LSTM Layers

The output from the final convolutional layer is reshaped into a format suitable for the LSTM layers. This layer flattens the spatial dimensions (time and frequency) while maintaining the temporal structure. After the convolutional and pooling layers, the data must be prepared for sequential processing by the LSTM layers. The reshape layer converts the spatial output from the convolutional layers into a sequence, where each time step contains the flattened feature maps. This is crucial to maintain the temporal order of the data when feeding it into the LSTM layers.

The model includes four LSTM layers, each with 128 units and return sequences = True to maintain the sequential nature of the data through all layers. LSTM layers are essential for capturing temporal dependencies in the speech data. Voice commands have sequential structures, and LSTM layers excel at modeling sequences over time. They "remember" relevant information from earlier time steps while processing current inputs, which is critical in speech processing. LSTMs handle long-term dependencies effectively, helping the model to understand how different frames (MFCCs over time) relate to each other to form coherent voice commands.

The model uses four LSTM layers to capture progressively higher-level temporal patterns. Each layer refines the temporal understanding of the sequence, allowing the model to better handle variations in how a command is spoken (e.g., different speeds, pauses, or inflections).

3.6.4 Dropout Layers and Time Distributed Dense Layers

Dropout layers are applied between the LSTM layers with dropout rates of 0.2 and 0.3. Dropout is used to prevent overfitting by randomly deactivating a fraction of the neurons during training. By doing this, the network is forced to learn more robust features and avoid relying too heavily on any single neuron. This is especially important for deep networks like this one, as overfitting can occur when the model becomes too specialized to the training data.

After the LSTM layers, the model applies several Time Distributed Dense layers with decreasing sizes (64, 32, 16, and 8 units), followed by ReLU activations. The Time Distributed layers apply dense (fully connected) layers to each time step independently, without losing the sequential structure of the data. This is necessary because after processing the input through LSTMs, each time step has its own feature vector, and a dense layer helps to refine the features for each frame independently. This step reduces the dimensionality while maintaining temporal relationships.

3.6.5 Flatten and Output Layer with Loss Function and Optimizer

The output of the Time Distributed layers is flattened, and a final Dense layer with a softmax activation function is applied to produce the final classification probabilities for the 5 commands. The Flatten layer converts the 3D output from the Time Distributed layers into a 1D vector for the final dense layers. This step is required before the final classification step, as the softmax layer expects a 1D input vector. The final dense layer with a softmax activation outputs the probabilities of the input belonging to one of the five voice command classes ("Samne Jao", "Pichone Jao", etc.). The softmax function converts the raw outputs into a probability distribution, making it easy to interpret the model’s prediction.

The model is compiled using the categorical cross-entropy loss function, which is appropriate for multi-class classification tasks. The Adam optimizer is used to adjust the model weights, offering an adaptive learning rate and efficient convergence.

3.6.6 Model Summary

- Input: Resized MFCCs with a shape of (250, 26, 1) (representing time frames and coefficients).
- Conv2D layers: Extract spatial features from the MFCCs.
- Max Pooling layers: Downsample feature maps to reduce complexity.
- LSTM layers: Capture temporal dependencies across time frames.
- Time Distributed Dense layers: Further refine features for each time step.
- Output layer: Classifies the input into one of the five commands with a softmax activation.

The complete model structure is shown in detail in the following fig. 3.5 with every step to step passing of the data.

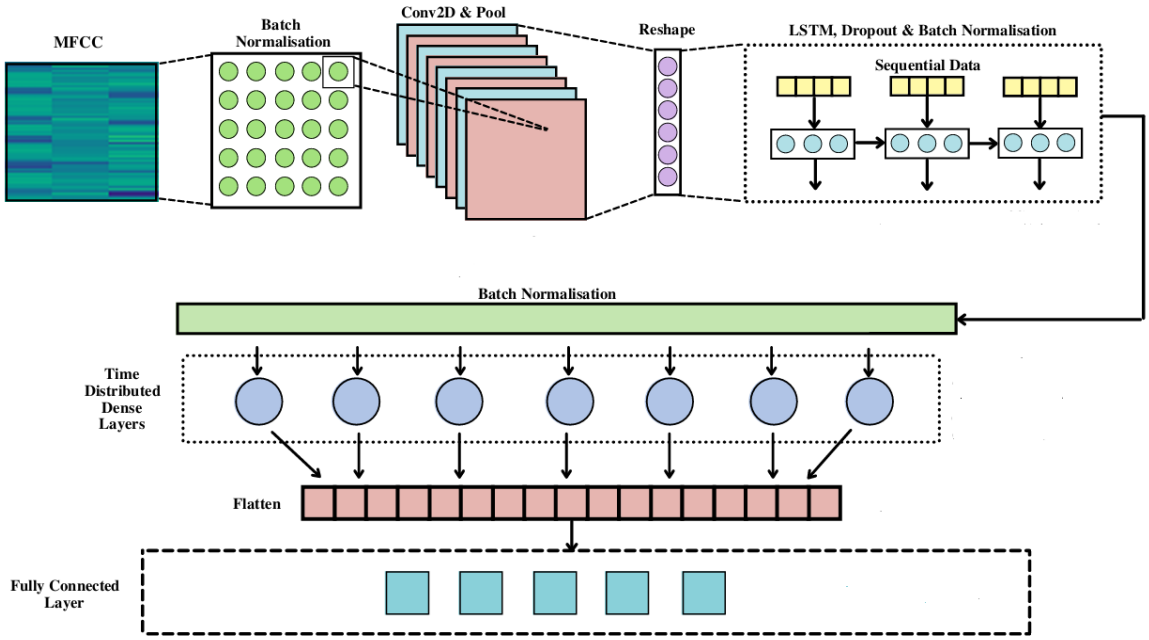


Fig. 3.5: The complete model layout

This CNN-LSTM hybrid architecture is specifically designed to handle the structure of speech data, using convolutional layers to capture the local spectral features and LSTM layers to learn the temporal dynamics of the voice commands. This makes it an efficient and accurate model for Bengali voice command recognition.

3.7 Model Training

The model was trained using a batch size of 20 over 100 epochs. The dataset, consisting of 1,500 samples, was split into an 80-20 ratio, with 1,200 samples for training and 300 samples for testing. During training, the 1,200 samples were further divided into 75-25 split for training (900 samples) and validation (300 samples).

- Batch size: 20 samples processed in each iteration.
- Optimizer: Adam optimizer, providing adaptive learning rates.
- Loss function: Categorical cross-entropy for multi-class classification.
- Validation: Ongoing evaluation during training to monitor performance and avoid overfitting.

Training also includes the use of batch normalization to stabilize learning and dropout layers to prevent overfitting, ensuring that the model learns robust features that generalize well to unseen data.

3.8 Testing and Validation

After training, the model was evaluated on a test set consisting of 300 samples, representing 20% of the total dataset. The test set was not used during training or validation, ensuring an unbiased assessment of the model's generalization capability.

The model's performance on the test set was measured using accuracy and loss metrics, calculated by comparing the predicted voice command labels with the true labels. This final evaluation step ensures that the model can effectively recognize Bengali voice commands in real-world scenarios and unseen data.

The model evaluated on a test set of 300 samples, achieved the following performance:

- Test loss: 1.5438
- Test accuracy: 81%

The model's accuracy in predicting each command was as follows:

- Samne Jao: 80%
- Pichone Jao: 80%

- Dane Jao: 76.67%
- Bame Jao: 78.33%
- Theme Jao: 90%

These results indicate strong performance across all commands, with the highest accuracy for "Theme Jao" at 90%, demonstrating the model's effectiveness in real-world voice command recognition.

Fig 3.6 shows the entire process of training and testing the dataset from start to finish with every step illustrated properly.

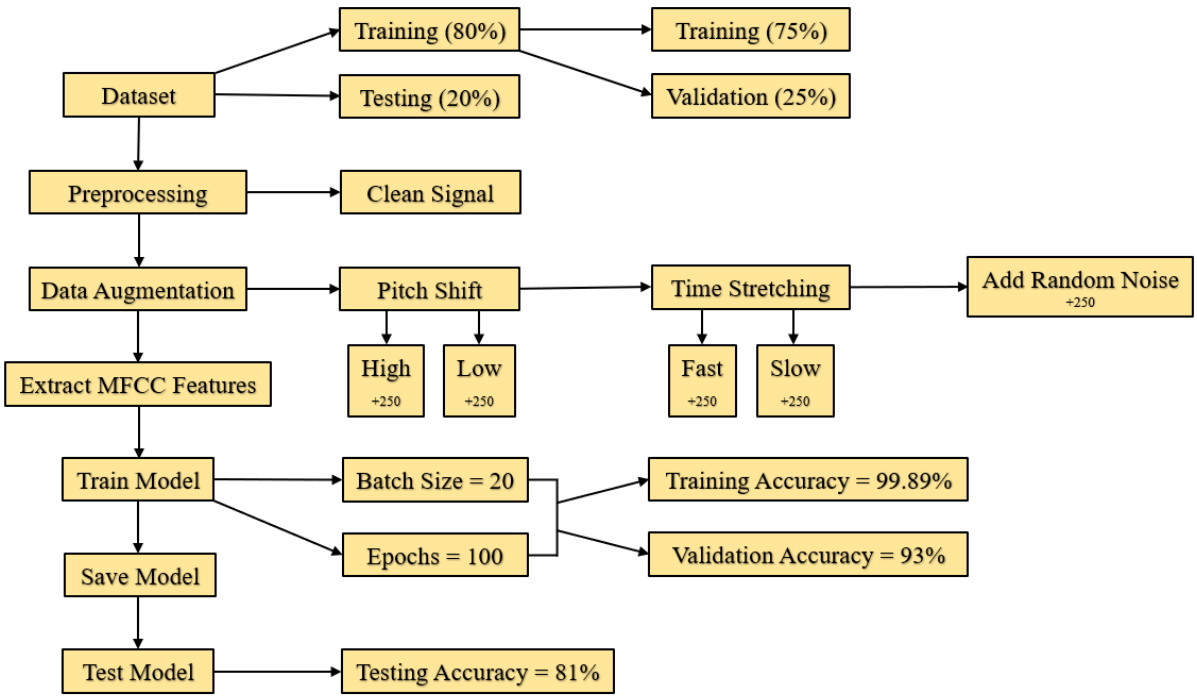


Fig. 3.6: The complete training and testing phase

CHAPTER IV

Results and Analysis

4.1 Implementation

The implementation of the thesis centers on deploying the trained CNN-LSTM model to recognize Bengali voice commands. The model is loaded from a previously saved keras file along with the corresponding label encoder for decoding predicted labels. The model is designed to classify five Bengali voice commands used for controlling a prototype robotic vehicle in an offline environment.

4.1.1 Loading The Model and Encoder

The first step involves loading the pre-trained model and label encoder. The model architecture and weights are retrieved from the saved keras file, and the label encoder is used to map predicted numeric outputs back into the original Bengali voice commands.

4.1.2 Data Preprocessing and MFCC Extraction

To evaluate the model on new data, a set of test audio files is loaded from a directory. Each audio file undergoes preprocessing to extract MFCC (Mel-Frequency Cepstral Coefficients) features, which represent the spectral characteristics of the audio signal. The extracted MFCCs are resized to a fixed shape of 250-time frames and 26 coefficients to match the input shape used during training.

In detail, MFCC extraction is performed using the python speech features library. The following steps are carried out:

- Load the audio signal and sampling rate.
- Extract MFCC features with 26 filter banks and 26 cepstral coefficients, using a frame size appropriate for speech signals.
- Resize the MFCC array to a shape of (250, 26) to ensure uniformity across inputs.

4.1.3 Model Evaluation

The model is evaluated on the test set, which is one-hot encoded using the same label encoder as during training. The evaluation metrics, including test accuracy and test loss, are calculated. The model achieved a test accuracy of 81%, showing its ability to correctly classify the Bengali voice commands.

Additionally, the binary accuracy for each individual command (e.g., Samne Jao, Pichone Jao) is calculated to provide insights into which commands the model recognizes most reliably. For example, "Theme Jao" achieved the highest individual accuracy of 90%, indicating strong performance for this command.

4.1.4 Prediction and Labeling

The model is used to predict labels for each test example, and the predicted labels are compared with the actual test labels to evaluate overall performance. The predictions are visualized using bar charts, showing the accuracy for each individual command, and binary results are plotted to reflect correct or incorrect predictions for each test case.

4.1.5 Real-Time Prediction and Live Audio

To further test the model's applicability, a function is implemented to predict voice commands from live audio. Audio is recorded using a microphone, preprocessed to clean and extract MFCC features, and then passed to the model for real-time prediction. The predicted command is displayed to the user, demonstrating the model's effectiveness in a live, real-time environment.

Additionally, the model's ability to handle noisy inputs is enhanced using an envelope-based noise reduction technique, which removes background noise from the input signal based on its energy profile. This preprocessing step ensures cleaner input, making the model more robust in real-world scenarios with background noise.

4.1.6 Real-Time Testing

A function allows the user to record audio commands live via a microphone. The recorded signal is passed through the preprocessing pipeline (including noise reduction and MFCC extraction) before being input to the model for prediction. This demonstrates the system's

ability to function offline and in real-time for controlling a vehicle with voice commands like “Samne Jao” or “Theme Jao”.

Fig 4.1 shows the entire process of real time testing from start to finish with every step illustrated properly.

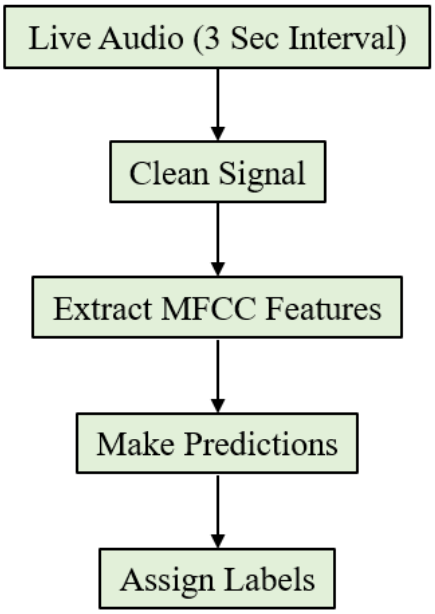


Fig. 4.1: The real time testing phase

The implementation of this Bengali voice command recognition system demonstrates its ability to accurately classify and respond to spoken commands, both from pre-recorded audio and real-time inputs. By running the model on a Raspberry Pi 4 and integrating it with a prototype robotic vehicle, the system can function in an offline environment, making it a practical solution for applications where internet access is unavailable. The model's robustness to noisy inputs and real-time voice recognition further enhances its practicality for real-world use cases.

4.2 Comparative Analysis with Other Models

CSVC-Net [1] achieved a 92.08% accuracy but focused on code-switched Bengali-English commands, which differs from the pure Bengali command set used in this thesis. Additionally, their approach required a large dataset, which was not available in this study. In contrast, our model's strength lies in its adaptability to smaller datasets and its offline

functionality.

IoT-Based Voice Controlled Vehicle [3] relied on Google Assistant and cloud computing, achieving success in controlling a robotic vehicle, but it was heavily dependent on the Google API system and an internet connection. Our model, by comparison, operates entirely offline, making it more practical for environments where internet connectivity is unreliable.

The Voice Controlled Robotic Car [4] used Bluetooth and mobile applications to process voice commands. It achieved high accuracy but relied on English-based commands and voice APIs, limiting its applicability in Bengali-speaking contexts. Our model is custom-built for Bengali, providing an inclusive solution.

Bangla Short Speech Commands Recognition [14] utilized CNN architectures to recognize short Bengali commands, achieving strong results. However, the use of raw audio files in their CNN model may not capture the sequential nature of speech as effectively as our CNN-LSTM hybrid, which is specifically designed to handle sequential time-series data like speech.

4.3 Robustness in Handling Defective Inputs

One of the significant strengths of this model lies in its ability to handle variations and defects in input speech. The data augmentation techniques applied during training (such as pitch shifts and time-stretching) helped the model generalize well to unseen variations. The model consistently provided correct labels even when users made slight errors in pronunciation, such as confusing "Samne Jao" with "Sane Jao" or "Bame Jao" with "Bane Jao." This robustness makes the system reliable for real-world applications, where perfect pronunciation cannot always be guaranteed.

4.4 Error Analysis

Though the overall performance was strong, certain commands, such as "Dane Jao" (76.67%) and "Bame Jao" (78.33%), showed slightly lower accuracy. This may be due to phonetic similarity between these commands, leading to occasional misclassifications. However, the model still maintains a high degree of accuracy across all commands, with "Theme Jao" achieving the highest accuracy of 90%. Further refinements in data

augmentation or model architecture could help reduce these misclassification errors in future iterations.

4.5 Summary of Results

In comparison to other models, this model offers a lightweight, offline solution for Bengali voice command recognition, outperforming cloud-dependent models in terms of practical applicability in environments with limited internet access.

It exhibits high accuracy in command recognition, even with defective inputs and noise, making it a reliable choice for real-world applications such as robotic control or hands-free vehicular operation.

The model presented in this thesis strikes a balance between performance and practicality, offering a robust offline voice recognition system specifically tailored for Bengali voice commands. While models like CSVC-Net [1] achieve higher accuracy, they focus on different applications (code-switching), and many rely on cloud-based systems. Our model, in contrast, is uniquely designed for an offline environment and performs exceptionally well even with limited data, making it a practical and efficient solution for voice-controlled applications in Bengali-speaking regions.

CHAPTER V

Conclusion

5.1 Summery

This thesis presented the development of a Bengali voice command recognition system based on a CNN-LSTM hybrid model. The model was designed to classify five essential vehicular commands: Samne Jao, Pichone Jao, Dane Jao, Bame Jao, and Theme Jao. It achieved a test accuracy of 81% and demonstrated high resilience to input variations and noise through data augmentation. The system was designed for offline use on Raspberry Pi, allowing robust, low-latency performance without reliance on cloud-based services. The model's effectiveness was validated even under non-ideal conditions such as distorted speech inputs.

5.2 Conclusive Remarks

- Developed a lightweight, custom-built model optimized for offline voice command recognition in Bengali.
- Leveraged data augmentation (pitch shift, time stretching, noise addition) to expand the training dataset and improve model robustness.
- Demonstrated the system’s ability to generalize to noisy environments and handle slight pronunciation errors (e.g., “Sane Jao” for "Samne Jao").

5.3 Limitations

- **Limited Dataset Size:** The dataset consisted of 1,500 augmented samples, which, while effective, may not represent the full variety of real-world speech patterns and environmental noise.
- **Confusion Between Similar Commands:** Commands with similar phonetic characteristics (e.g., "Dane Jao" and "Bame Jao") had lower accuracy compared to others, indicating a need for further refinement in distinguishing these nuances.

- **Hardware Constraints:** Although the model is designed to run on a Raspberry Pi 4, the real-time performance on low-power hardware could be impacted by latency, particularly in noisy or dynamic environments.

5.4 Recommendations and Future Work

- **Dataset Expansion:** To improve model robustness, collecting a larger, more diverse dataset from a wider range of speakers and environments will help generalize the system to a broader range of speech patterns and acoustic conditions.
- **Refining Phonetic Distinctions:** Further work could focus on refining the model to better differentiate between phonetically similar commands, possibly by integrating advanced phonetic modeling techniques or additional acoustic features.
- **Hardware Optimization:** Future efforts should focus on optimizing the model for real-time performance on low-power hardware, such as Raspberry Pi or similar embedded devices, to ensure smooth, latency-free command recognition.
- **Language and Command Expansion:** The system can be expanded to recognize a larger set of Bengali commands or support code-switching between Bengali and English to broaden its usability, especially for bilingual users. Additionally, the model could be adapted for use in other applications, such as smart home systems or assistive technologies for the physically impaired.
- **Improved Noise Handling:** Incorporating real-world background noise profiles into the augmentation process can further improve the system's robustness, enabling better performance in noisy environments.

This thesis establishes a strong foundation for future work in developing offline, language-specific voice recognition systems. With further advancements in data, model architecture, and real-world testing, this system has the potential for broader applications beyond vehicular control, including home automation, assistive technologies, and smart devices in Bengali-speaking regions.

References

[1] A. Yasmeen, F. I. Rahman, S. Ahmed and M. H. Kabir, "CSVC-Net: Code-Switched Voice Command Classification using Deep CNN-LSTM Network," *2021 10th International Conference on Informatics, Electronics & Vision (ICIEV)*, Kitakyushu, Japan, 2021, pp. 1-8, doi: 10.1109/ICIEVicIVPR52578.2021.9564183.

[2] G. K. Berdibayeva, A. N. Spirkin, O. N. Bodin and O. E. Bezborodova, "Features of Speech Commands Recognition Using an Artificial Neural Network," *2021 Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, Yekaterinburg, Russia, 2021, pp. 0157-0160, doi: 10.1109/USBEREIT51232.2021.9455111.

[3] M. Gupta, R. Kumar, R. K. Chaudhary and J. Kumari, "IoT Based Voice Controlled Autonomous Robotic Vehicle Through Google Assistant," *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2021, pp. 713-717, doi: 10.1109/ICAC3N53548.2021.9725526.

[4] S. Chakraborty, N. De, D. Marak, M. Borah, S. Paul and V. Majhi, "Voice Controlled Robotic Car Using Mobile Application," *2021 6th International Conference on Signal Processing, Computing and Control (ISPCC)*, Solan, India, 2021, pp. 1-5, doi: 10.1109/ISPCC53510.2021.9609396.

[5] M. R. Sultan, M. M. Hoque, F. U. Heeya, I. Ahmed, M. R. Ferdouse and S. M. A. Mubin, "Adrisya Sahayak: A Bangla Virtual Assistant for Visually Impaired," *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, DHAKA, Bangladesh, 2021, pp. 597-602, doi: 10.1109/ICREST51555.2021.9331080.

[6] S. M. Islam, M. F. A. Houya, S. M. Islam, S. Islam and N. Hossain, "Adheetee: A Comprehensive Bangla Virtual Assistant," *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, Dhaka, Bangladesh, 2019, pp. 1-6, doi: 10.1109/ICASERT.2019.8934903.

[7] M. S. Hossain Shawon *et al.*, "Voice Controlled Smart Home Automation System Using Bluetooth Technology," *2021 4th International Conference on Recent Trends in Computer Science and Technology (ICRTCST)*, Jamshedpur, India, 2022, pp. 67-72, doi: 10.1109/ICRTCST54752.2022.9781967.

[8] M. S. A. Chowdhury and M. F. Khan, "Linear predictor coefficient, power spectral analysis and two-layer feed forward network for bangla speech recognition," *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, Pondicherry, India, 2019, pp. 1-6, doi: 10.1109/ICSCAN.2019.8878709.

[9] P. Gawade and S. K. P, "In-Vehicle Speech Command Operated Driver Assist System for Vehicle Actuators Control using Deep Learning Techniques," *2021 5th International Conference on Computer, Communication and Signal Processing*

(ICCCSP), Chennai, India, 2021, pp. 262-267, doi: 10.1109/ICCCSP52374.2021.9465511.

[10] N. Sadeq, S. Ahmed, S. S. Shubha, M. N. Islam and M. A. Adnan, "Bangla Voice Command Recognition in end-to-end System Using Topic Modeling based Contextual Rescoring," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, 2020, pp. 7894-7898, doi: 10.1109/ICASSP40776.2020.9053970.

[11] M. Islam, M. S. Hossain and N. Akhter, "Hate Speech Detection Using Machine Learning In Bengali Languages," *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2022, pp. 1349-1354, doi: 10.1109/ICICCS53718.2022.9788344.

[12] R. L. Khan, D. Priyanshu and F. S. Alsulaiman, "Implementation of Human Voice Controlled Robotic Car," *2021 10th International Conference on System Modeling & Advancement in Research Trends (SMART)*, MORADABAD, India, 2021, pp. 640-646, doi: 10.1109/SMART52563.2021.9676319.

[13] A. Azarang, J. Hansen and N. Kehtarnavaz, "Combining Data Augmentations for CNN-Based Voice Command Recognition," *2019 12th International Conference on Human System Interaction (HSI)*, Richmond, VA, USA, 2019, pp. 17-21, doi: 10.1109/HSI47298.2019.8942638.

[14] S. Ahmed Sumon, J. Chowdhury, S. Debnath, N. Mohammed and S. Momen, "Bangla Short Speech Commands Recognition Using Convolutional Neural Networks," *2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, Sylhet, Bangladesh, 2018, pp. 1-6, doi: 10.1109/ICBSLP.2018.8554395.

[15] D. Gupta, E. Hossain, M. S. Hossain, K. Andersson and S. Hossain, "A Digital Personal Assistant using Bangla Voice Command Recognition and Face Detection," *2019 IEEE International Conference on Robotics, Automation, Artificial-intelligence and Internet-of-Things (RAAICON)*, Dhaka, Bangladesh, 2019, pp. 116-121, doi: 10.1109/RAAICON48939.2019.47.

[16] S. T. Yeasin Ramadan, T. Sakib, M. A. Rahat, M. Mushfique Hossain, R. Rahman and M. M. Rahman, "An Integrated Embedded System Towards Abusive Bengali Speech and Speaker Detection Using NLP and Deep Learning," *2022 25th International Conference on Computer and Information Technology (ICCIT)*, Cox's Bazar, Bangladesh, 2022, pp. 698-703, doi: 10.1109/ICCIT57492.2022.10054785.

[17] S. A. Rahat, A. Imteaj and T. Rahman, "An IoT based Interactive Speech Recognizable Robot with Distance control using Raspberry Pi," *2018 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, Chittagong, Bangladesh, 2018, pp. 480-485, doi: 10.1109/ICISET.2018.8745656.