Indian Language Detectionfrom Audio Data

|  |  |  |
| --- | --- | --- |
|  |  |  |

Mannepalli Hari Virinchi   
Department of Computer ScienceAmrita School of Engineering,   
Amrita Vishwa Vidyapeetham,Bengaluru, India.  
[BL.EN.U4AIE21077@bl.students.amrita.edu](mailto:BL.EN.U4AIE21077@bl.students.amrita.edu)

Bathala Somasekhar  
Department of Computer Science Amrita School of Engineering,  
Amrita Vishwa Vidyapeetham, Bengaluru, India.  
[BL.EN.U4AIE21020@bl.students.amrita.edu](mailto:BL.EN.U4AIE21020@bl.students.amrita.edu)

Vaan Amuthu Elango  
Department of Computer Science Amrita School of Engineering,   
Amrita Vishwa Vidyapeetham, Bengaluru, India.  
[e\_vaanamuthu@blr.amrita.edu](mailto:e_vaanamuthu@blr.amrita.edu)

**Abstract- The purpose of this project is to develop an Indian language identification system using audio data. By employing audio processing and machine learning the system precisely groups spoken segments of the 10 Indian languages. Quality checks guarantee the dataset authenticity, MLP is able to get impressing accuracy. The system is promising as it could be applied to voice-controlled devices and multilingual speech processing applications**

***Keywords:Indian language identification, audio processing, machine learning, Multi-Layer Perceptron (MLP), dataset authenticity, quality checks, voice-controlled devices, multilingual speech processing.***

# INTRODUCTION

The communication language plays are an inevitable element that helps in building the bridge which ties people to the surrounding. Linguistic-scape in India cover the interesting mixture of more than 700 languages/dialects; being able to mark them and to grasp it accurately is obligatory. Yet, the conventional methods of language identification do not usually include the intricacies of spontaneous dialectic communication as a factor.

In this project, an audio-based language identification system with advanced automation for Indian languages is being devised to solve the issue of language identification. The phase of innovations in digital tools and machine learning algorithms became a marker in language processing where the language recognition systems reach the highest level possible.  
  
Dataset performed as a foundation of the project wherein it was audio samples library which are 5 seconds long and was spoken in 10 different Indian languages. Such samples were wisely selected from the video clips existing on YouTube, which, therefore, produced a more inclusive and representative dataset. The mentioned point should be taken into account but the fact is that no audio samples or video sources have been owned by the project owner and all the data used is ethically collected following specific rules and regulations.

The primary goal of this project is to use audio processing methods together with machine learning algorithms to discriminate spoken fragments of Indian languages in a precise way. By means of a complete workflow including data preprocessing, feature extraction, model training, and evaluation, the project would like to bring the level of robust language identification up to the mark. Furthermore, strict quality controls are deployed to prevent errors and biases that can impact the accuracy of the dataset and consequently, model performance.  
  
In the sections below, we describe the technical details of the project, clarifying the steps of audio processing, feature extraction and model training. In addition, the effectiveness of the developed Indian language identification system is proved along with the various domains where it can be applied to, like voice-controlled devices and multilingual speech processing tools.

# LITERATURE SURVEY

The work presented in this paper highlights the existing research gap in optimizing feature sets for spoken language identification, particularly in the context of Indian languages. While previous studies have primarily focused on feature extraction and classification models, little attention has been given to feature selection techniques to reduce model complexity and training time. The proposed hybrid meta-heuristic feature selection method, integrating Binary Bat Algorithm with Late Acceptance Hill-Climbing, fills this gap and demonstrates superior performance compared to standard methods, offering a promising avenue for future research in the field.[1].

The work conducted in this comprehensive review sheds light on the significant advancements and challenges in Indian spoken language recognition (LID) research. It underscores the importance of adapting smart technologies to accommodate the linguistic diversity of India's population. Despite notable progress, challenges such as low-resource availability and mutual influences among languages persist, impacting the accuracy of LID systems, particularly in distinguishing closely related language pairs. The review provides a valuable resource for researchers and enthusiasts, offering insights into the current state of research and suggesting future directions for advancing Indian LID technology.[2].

The work presented in this paper emphasizes the application of deep learning architectures, particularly convolutional neural networks (CNNs), for spoken language identification (SLID) tasks. The study explores the effectiveness of using spectrogram images derived from audio files for language detection, achieving high accuracy rates. Additionally, the paper compares the performance of CNNs with traditional methods like Bernoulli Naïve Bayes and pretrained models, demonstrating the superiority of deep learning approaches. Further avenues for improvement are suggested, including data augmentation techniques and exploring alternative feature extraction methods to enhance SLID accuracy in real-world scenarios.[3].

The work presented in this paper highlights the significance of spoken language identification (SLID) as a crucial step in multilingual speech recognition systems. It introduces a novel approach utilizing MFCC-based time series features augmented with noise to enhance model robustness. The proposed model achieves high accuracy rates on three standard datasets, showcasing the effectiveness of the extracted features in capturing language-specific characteristics. While the model outperforms previous works, the computational expense of feature extraction is acknowledged, suggesting potential avenues for more efficient techniques and exploring end-to-end learning with recurrent neural networks (RNNs) or long short-term memory (LSTM) models'.[4].

The work provided in this paper underscores the critical importance of automatic spoken language identification (SLID) for various applications in the modern era. The proposed FuzzyGCP architecture introduces a novel approach to SLID, utilizing a deep learning ensemble consisting of Deep Dumb Multi Layer Perceptron (DDMLP), Deep Convolutional Neural Network (DCNN), and Semi-supervised Generative Adversarial Network (SSGAN), combined with Ensemble learning using Choquet integral. Evaluation on diverse datasets, including both Indic and foreign languages, demonstrates high F1-scores, showcasing the robustness and versatility of the proposed model. Future research directions include exploring feature selection algorithms, less computationally intensive architectures, and sequential models like Gated Recurrent Units (GRUs) and LSTMs, as well as further analysis of x-vector and i-vector based models to enhance multi-lingual SLID systems for various applications like tele-medicine and automatic translation switching frameworks.[5].

The literature survey in this paper underscores the challenges in developing speech technology applications for low-resource languages (LRL), particularly in the context of Eastern and Northeastern (E&NE) Indian languages. It emphasizes the scarcity of proper speech corpora for these languages, hindering large-scale development efforts. The paper addresses this gap by detailing the creation process of an LRL corpus comprising sixteen rarely studied languages, providing a valuable resource for research and system development in the field. Experiments on speaker and language identification using various spectral features and classifiers demonstrate promising baseline performance, highlighting the importance of such initiatives in advancing speech technology for underrepresented languages.[6].

# METHODOLOGY

**Lab 1**

Loading and Visualization:The audio file is imported and loaded into the Python environment using the librosa library's load function, resulting in two main variables: 'y', containing the audio data, and 'sr', representing the sampling rate. Utilizing the matplotlib library, the waveform of the entire audio signal is plotted, providing a graphical representation of the amplitude variation over time.

Segmentation and Labeling: Through manual inspection, segments of the audio containing spoken words and periods of silence are identified. These segments are then marked on the waveform plot using colored spans, each labeled with descriptive characters such as 'A', 'I', 'in', 'Speech', and 'Processing', aiding in visual interpretation.

Duration and Decibel Calculation: The duration of the audio signal in seconds is computed by dividing the length of the audio data array by the sampling rate ('sr').

To assess the magnitude range of the audio signal, the decibel value is calculated, revealing the maximum amplitude present within the signal.

Deconstruction and Visualization: The audio signal is deconstructed into various intervals based on specified time ranges. Each segment is individually plotted to visualize its waveform characteristics, facilitating the understanding of unique variations across different portions of the audio.

Silence Trimming:Usinglibrosa's trim function with a specified threshold of 20 dB, regions of silence within the audio signal are identified and subsequently trimmed. The trimmed audio segments are plotted as waveforms to demonstrate the effect of silence trimming on the signal's amplitude and temporal profile.

Resampling: To explore the impact of different sampling rates on the audio signal, resampling is performed using librosa's resample function, targeting desired rates such as 40,000 Hz and 8,000 Hz. Waveforms of the resampled audio are plotted to visualize alterations in signal characteristics induced by the resampling process.

Spectrogram Analysis:Employing Short-Time Fourier Transform (STFT), spectrograms are computed to reveal the frequency content of the audio signal over time. Both conventional spectrograms and mel spectrograms are generated and plotted, offering detailed insights into the spectral features of the audio signal.

**Lab 2**

Data Extraction and Preprocessing:The original speech signal is extracted from the "soma.wav.ogg" file using the wavfile.read function from scipy.If the audio is stereo, it is converted to mono by averaging the channels.

Derivative Computation:The first derivative of the speech signal is computed using the Finite Difference Method (FDM), resulting in another audio file named "first\_derivative.wav".

Both the original speech signal and its derivative are plotted using matplotlib, with the original signal displayed on the top subplot and the derivative on the bottom subplot, illustrating their amplitude variations over time.

Zero Crossing Analysis: The first derivative signal is loaded into memory again using librosa.load.

Zeros are detected in the first derivative signal using librosa'szero\_crossingsfunction.Distances between consecutive zero crossings are calculated, representing speech and silence intervals based on a given threshold.The code interprets the first derivative signal overlaid by speech and silence areas, identifying zero crossings.

Mean Length Calculation: Mean lengths of zero crossings between continuous speech and silence are computed, providing insights into the periodic properties of spoken and silence segments in the audio signal.

Word Length Comparison:Speech lengths of two individuals ("Harivirinchi" and "Somasekhar") are evaluated by accessing their respective audio files and determining the duration of each sample.Bar plots are created to represent and compare the speech lengths between the two individuals.

Individual Word Visualization:Theplot\_audio\_signals function is utilized to diagram the audio signals for individual words spoken by "Harivirinchi" and "Somasekhar".

This function generates plots representing the amplitude of audio signals with their titles as axes, allowing visual inspection of the pronunciation patterns of words.

Conclusion: The code conducts various analyses and comparisons of speech signals, including derivative computation, identification of speech and silence regions, comparison of word lengths between individuals, and visual representations of individual words. These analyses provide interpretations regarding the temporal features and pronunciation patterns of words in speech.

**Lab 3**

Silence Removal using librosa.effects.trim(): The librosa.effects.trim() function is employed initially to eliminate silent portions from the beginning and end of the recorded speech signal. This ensures that only meaningful audio segments containing speech are retained for further analysis.

Segmentation with librosa.effects.split(): The recorded speech signal is segmented using the librosa.effects.split() function based on detected silences.Different thresholds (top\_db values) are experimented with to observe their effects on the quality of segmentation. Each segment is identified and marked within the original audio waveform for visualization.

Custom Silence Detection Algorithm: Inspired by the paper "Silence Detection and Removal Method Based on the Continuous Average Energy of Speech Signal," a custom silence detection algorithm is implemented. This algorithm calculates the energy of the signal and compares it against a predefined threshold to identify silent regions within the audio. The goal is to explore and compare different approaches for silence detection and segmentation in speech signals.

**Lab 4**

Amplitude Spectrum Observation using Fast Fourier Transform (FFT): The amplitude spectrum of the speech signal is observed by applying Fast Fourier Transform (FFT) using numpy.fft.fft(). This technique provides insight into the distribution of amplitudes across different frequencies in the speech signal.

Signal Reconstruction Evaluation: The fidelity of the signal reconstruction process is assessed by comparing the signal derived from computing the inverse Fourier transform using numpy.fft.ifft() with the original speech signal. By comparing the historical and reconstructed signals in the time domain, the adequacy of the reconstruction process is evaluated.

Frequency Domain Analysis of Speech Signal: Specific portions of the speech signal are identified and featured for analysis, converting time-dependent signals into the frequency domain to discern corresponding frequencies.The Fourier spectrum is obtained by plotting it through a rectangular window function applied to the speech signal, emphasizing localized spectral features in specific time windows.

Short-Time Fourier Transform (STFT) and Spectrogram Analysis: Short-Time Fourier Transform (STFT) is computed using librosa.stft() to analyze the speech signal in the time-frequency domain. This technique facilitates understanding the time-varying nature of the frequency content within the speech signal. Additionally, a spectrogram of the speech signal is generated using scipy.signal.spectrogram(), providing a time-frequency distribution of spectral intensity for examining time-varying spectral characteristics within the speech signal.

Visualization Techniques: Various visualization techniques, including plotting amplitude spectrum, time-domain signals, spectrograms, and frequency components, are employed to provide visual representations of different aspects of the speech signal.

**Lab 5**

Library Import and Preprocessing: The necessary libraries such as numpy, librosa, and matplotlib are imported. The audio signal is loaded using librosa, ensuring the signal is in the appropriate format for analysis.

Spectral Domain Analysis: The audio signal is transformed into the spectral domain using Fast Fourier Transform (FFT). The amplitude spectrum of the speech signal is visualized to understand the distribution of amplitudes across different frequencies.

Time Domain Analysis: The time-domain signal is reconstructed from the spectral components using Inverse Fast Fourier Transform (IFFT). The original and reconstructed signals are compared to evaluate the fidelity of the reconstruction process.

Spectrogram Generation: A spectrogram of the speech signal is generated using a windowed approach with Short-Time Fourier Transform (STFT).This spectrogram provides a visualization of the signal's frequency content over time.

Filtering Techniques: Various filters such as cosine, Gaussian, band-pass, high-pass, low-pass, Butterworth, band-stop, and peaking-filters are applied to the spectral components of the signal.

Filtered signals are obtained by applying these filters to the FFT results, and the resulting signals are stored as audio files.

The filtered signals are then either listened to or visualized to analyze the effects of each filter on the frequency components and temporal characteristics of the speech signal.

**Lab 6**

Library Import and Preprocessing:The necessary libraries such as librosa, matplotlib, and numpy are imported.The audio signal is loaded using librosa, ensuring the signal is in the appropriate format for analysis.

Vowel Analysis:The analyze\_vowel function is defined to analyze specific vowel segments within the audio file.

Vowel segments are extracted from the audio file based on the provided start and end times.Fast Fourier Transform (FFT) is applied to the vowel segment to obtain the frequency spectrum.The amplitude spectrum of the vowel sound is plotted to visualize the frequency content.

Consonant Analysis:The analyze\_sound function is defined to analyze specific consonant segments within the audio file.Consonant segments are extracted from the audio file based on the provided start and end times.FFT is applied to the consonant segment to obtain the frequency spectrum.The amplitude spectrum of the consonant sound is plotted to visualize the frequency content.

Silence and Non-Voiced Analysis:Potential silence and non-voiced portions of the audio file are identified and analyzed.FFT is applied to these segments to obtain the frequency spectrum.The amplitude spectrum of the suspected silent and non-voiced portions is plotted for examination.

Spectrogram Generation and Transition Detection:The generate\_spectrogram function is defined to generate a spectrogram of the entire audio signal.Mel-frequency cepstral coefficients (MFCCs) are computed to represent the spectrogram.The spectrogram is visualized, displaying the intensity of frequencies over time.Heuristic-based identification is performed to detect potential consonant-to-vowel transitions based on intensity changes.

**Lab-7:**

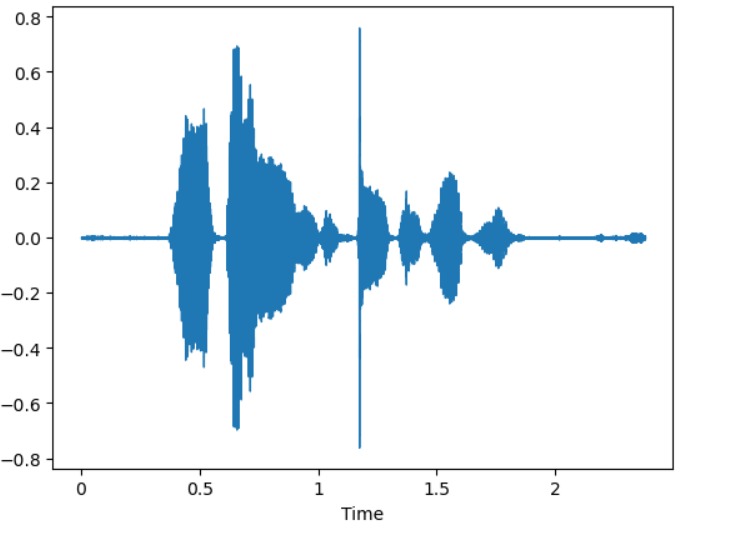
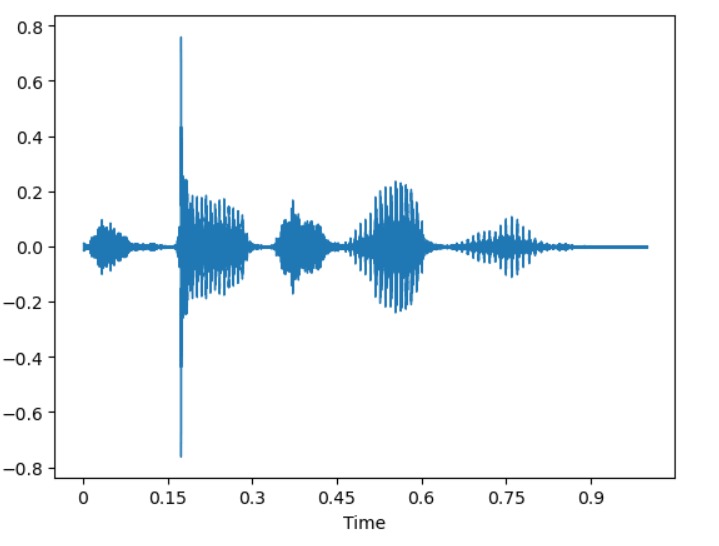
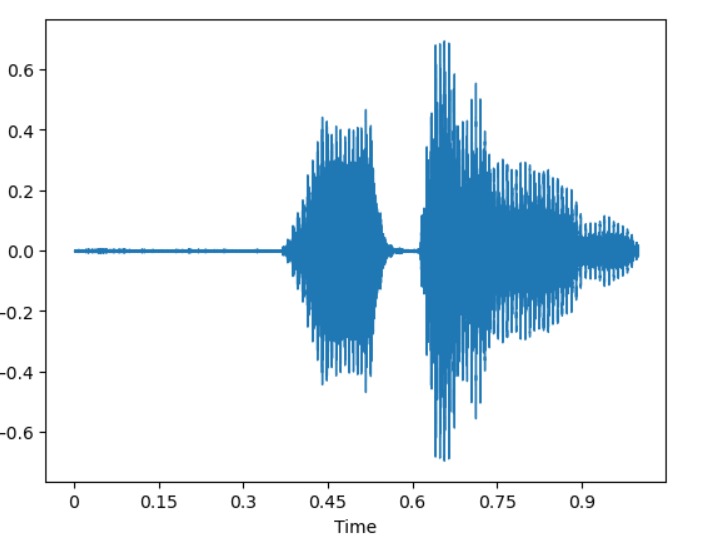
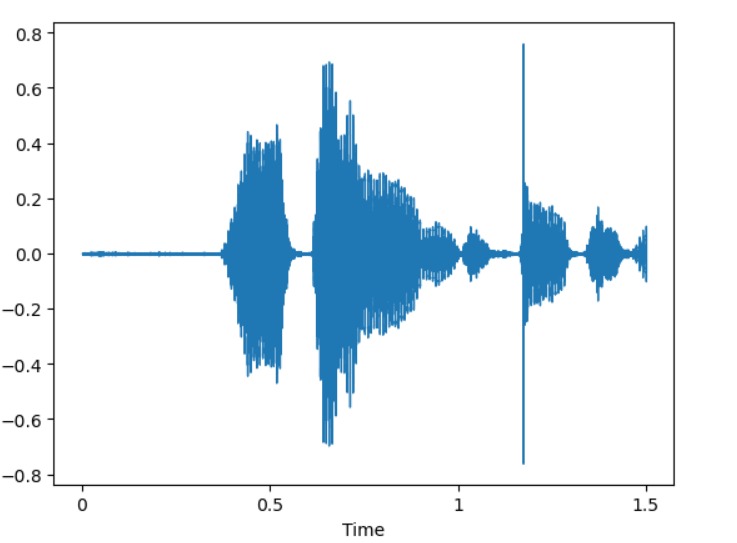
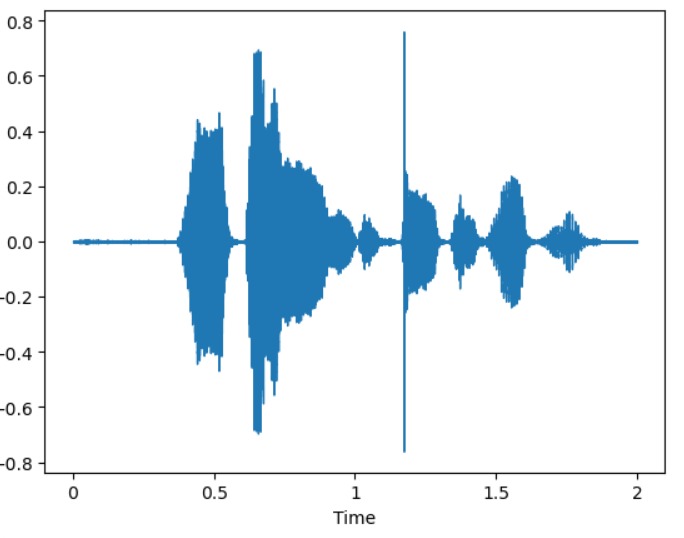
Audio features are a must for audio data that can be used to describe the traits that have been accepted as major in carrying out the operation. In this methodology, three types of features are extracted from the input audio files: STFT (Short-Time Fourier Transform), MFCCs (Mel-Frequency Cepstral Coefficients) and LPC (Linear Pre-dictive Coding. With the STFT function, you improve the audio signal's frequency representation over time, but MFCCs will capture spectral features and LPC coefficients serves as the model of the signal's linear predictable properties. Compared to the other network architecture, this one is capable of selecting these variable attributes and consequently to have access to multiple aspects of the sound, these characteristics allowing the model to learn robust representations for classification.  
  
After that, the extracted features have to be organized, ensured of eligibility and prepared for training of neural network models. The\_data\_process function is designed to meet this objective through using a list of audio file paths as an input and returning arrays of STFT, MFCC and LPC features, referring them to the vox\_data dataset. These data sets are structured coherently with the type of data networks can process for training purposes. Appropriate data processing contributes to the uniform arrangement of input data into the required format and also promotes the steady training process.  
This model strategy is the supervised Machine Learning, where the paired text with the traveler's review. The neural network used in this methodology are Long Short-term Memory (LSTM) and Bidirectional LSTM (BiLSTM). These models are the best option for the task to be performed to see that temporal dependencies are chosen particularly in the case of sequential data such as audio signals. I modeled each network using Keras with the backend of TensorFlow due to the Keras API allows us to describe the architecture. Both the LSTM and BiLSTM architectures commence from the layers of LSTM units which, in turn, are followed by dropout layers to respond to overfitting. Furthermore, we integrate dense layers into the classifier part of our models. This helps to learn complex patterns of the input data.

Before training, the data must be compiled with the configurations that ensure their parameters as required and evaluate themselves to determine their performance after training.  
Now that the models are developed, the next step is to train them using the audio data that has been transformed for the ideal format with already annotated labels. The process of training involves utilization of many batches of input data which are fed to models resulting in parameter-adjustment until the loss function is minimized as much as possible. The models are trained for a total of specified epochs (a single epoch is an iteration of the entire dataset) and with a batch size. Apart from the rest training data, which is additionally set aside for the validation as a metric to track the models' performance and to prevent their overfitting.  
  
The models could be given some forms of training after which they can be tested on unseen data to assess how well they have performed. Evaluation parameters like accuracy help us draw ad generalization ability of the frameworks that it can deal with new samples. However the proper results may necessitate fine-tuning of hyper parameters or adjustments of model architecture which will result in the improvement of performance. This repetitive practice of evaluating and fine-tuning contributes to the improvement of the models for instance, on the basis of initially collected samples and hence, this process ensures better classification accuracy. optimization of parameters as well as the evaluation metrics. In this approach the models are assembled with the adam optimized cross-entropy loss function (it's particularly suited in multi-class classification tasks) and accuracy as the evaluation metric. It sets the stage for the training by setting up the models in such a way that they will adapt

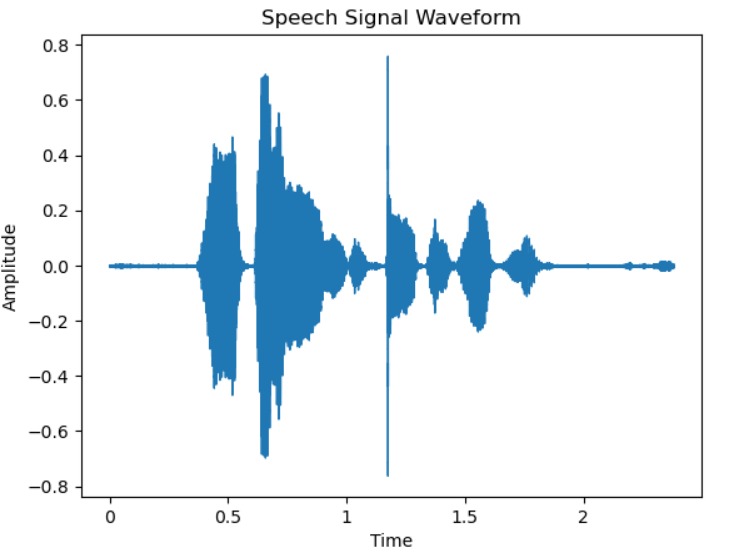
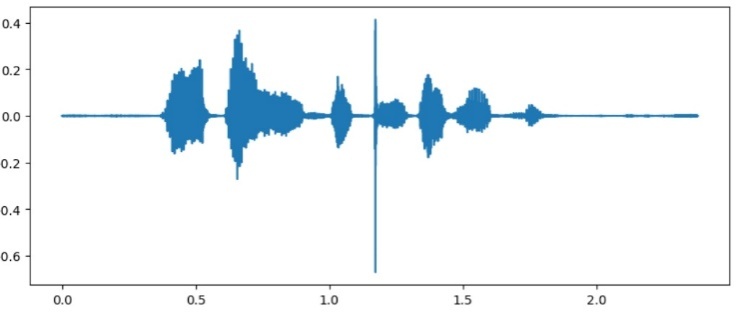
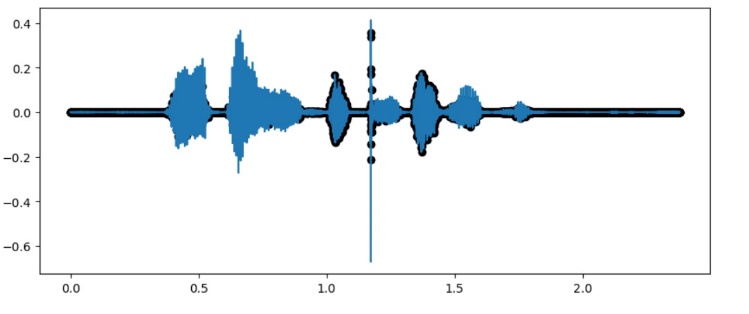
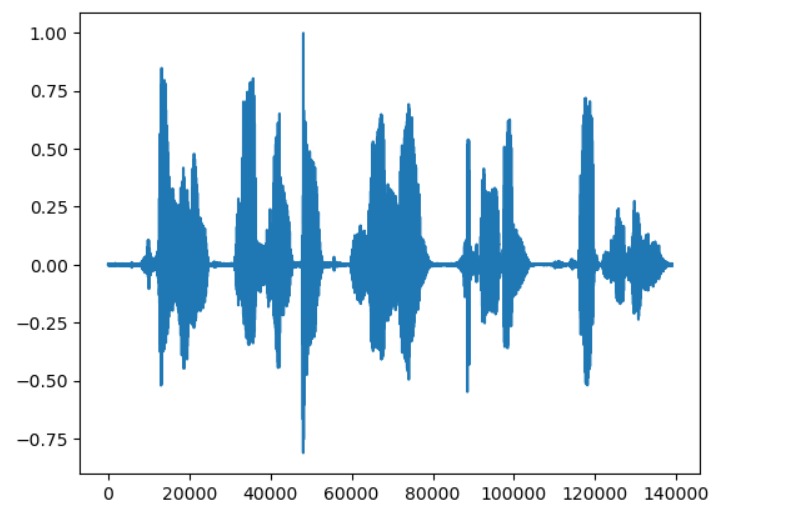
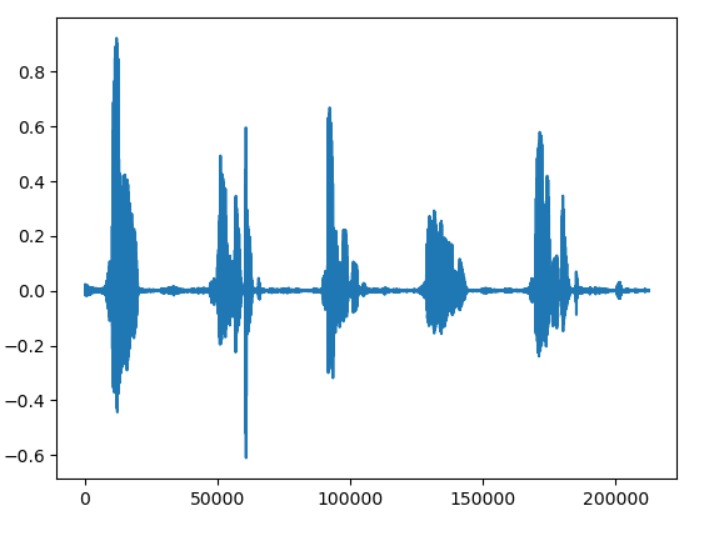
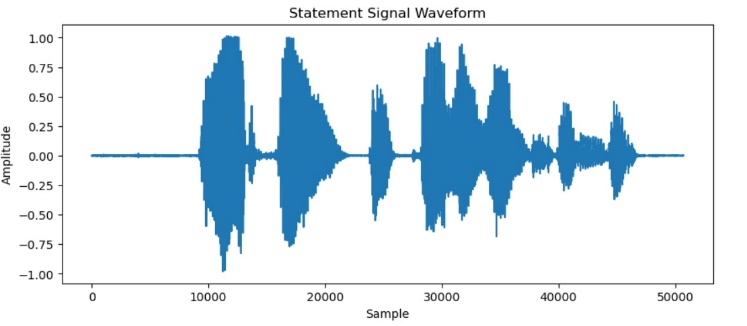
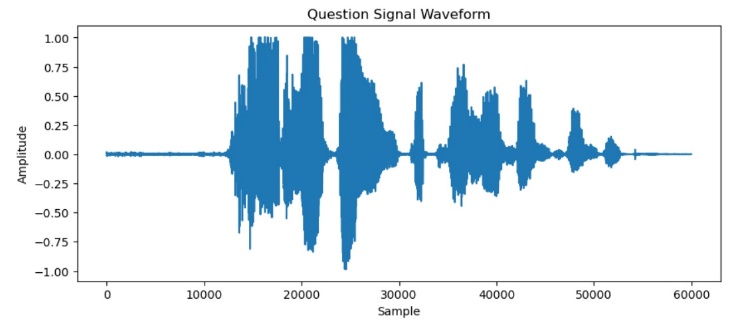
**Lab-8:**  
The first stage of the code deals with acquiring audio signals and deriving useful extracted features. It uses librosa lib to load audio files and performs a number of the signal processing operations. A more exact description of the process would be to say that it does the Short-Time Fourier Transform (STFT) and Mel-Frequency Cepstral Coefficients (MFCCs) calculation from the audio data. These characteristics, in turn, convey vital properties of the audio including the content of frequencies and the spectral envelope, thus, important for computer vision tasks like speech recognition.  
  
The MFCC values are extracted next and the data is ready for the LSTM model which handles the training. The MFCC matrix is transposed in order to meet the LSTM model's dimensional requirement and an extra dimension is adeded to represent the channel. With this transformation, the data is up to industry standards, and the network is prepared for learning.  
  
The necessary steps in the LSTM model architecture are defined subsequently. This layer comprise of Bidirectional LSTM with TimeDistributed Dense layer presenting ReLU as an activation function. This kind of regularization is done so that the model won‘t fit any data literally. Moreover, the generated probabilities are obtained as a result of application of a stack of Bidirectional LSTM layer with TimeDistributed Dense layer for output that is followed by softmax activation. This structure is appropriate as it relates to the sequential data like the speech signals, since it is capable of capturing the long-term randomness efficiently.  
  
LSTM model is compiled with the help of Adam optimizer and categorical crossentropy loss function. The optimizer ensures a low loss during training and classifies multi-class problems like speech recognition, the categorical crossentropy is the one to use. Precision is chosen, as the evaluation parameter to measure the model effectiveness during the training process.  
  
The next step of the code resolves the speech making and evaluation. It will convert an audio file by applying Google's speech recognition API and will split the phonemes for a word in the transcript. Utilizing the segmented phonemes, it represents speech to be played for the target word. The next step is to display the original and reconstructed signals simultaneously so that it can be visually inspected to identify the voice rebuilt via synthesis.

# RESULTS

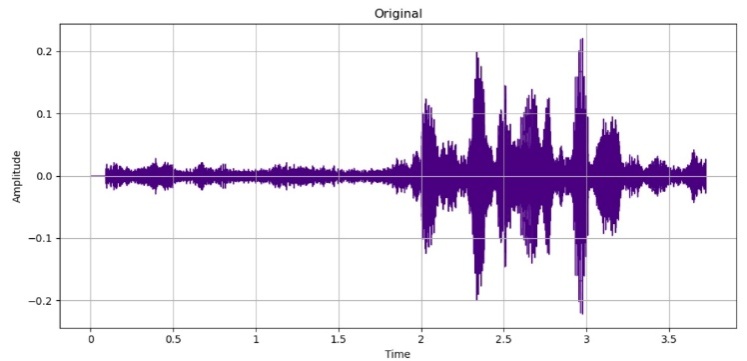
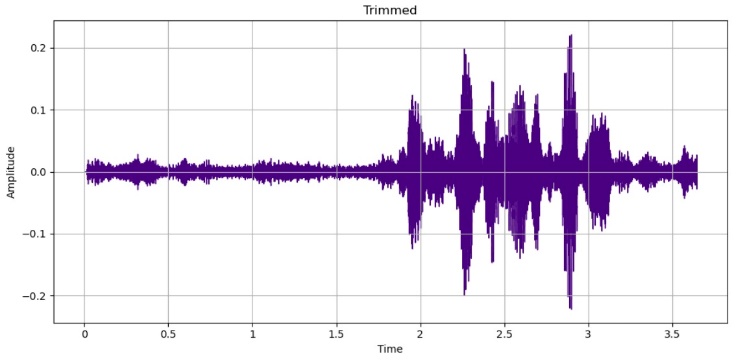
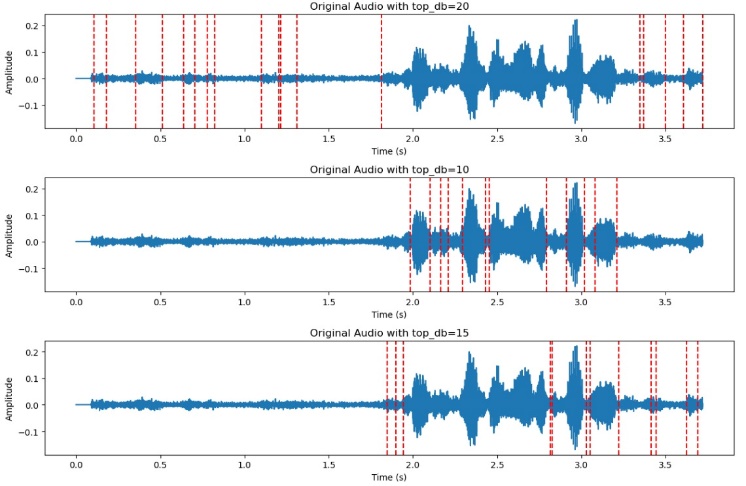
**Lab-1**

****

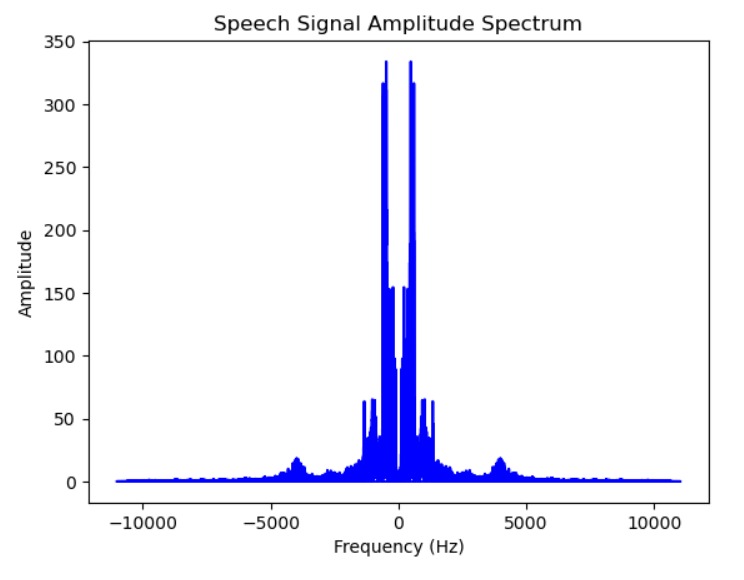
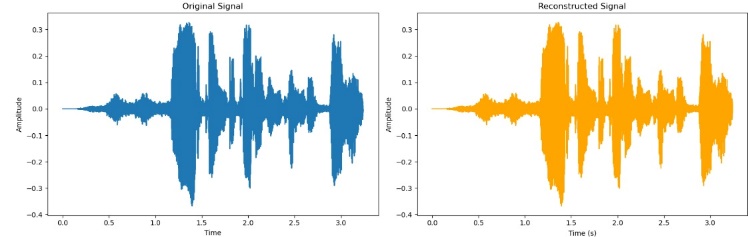
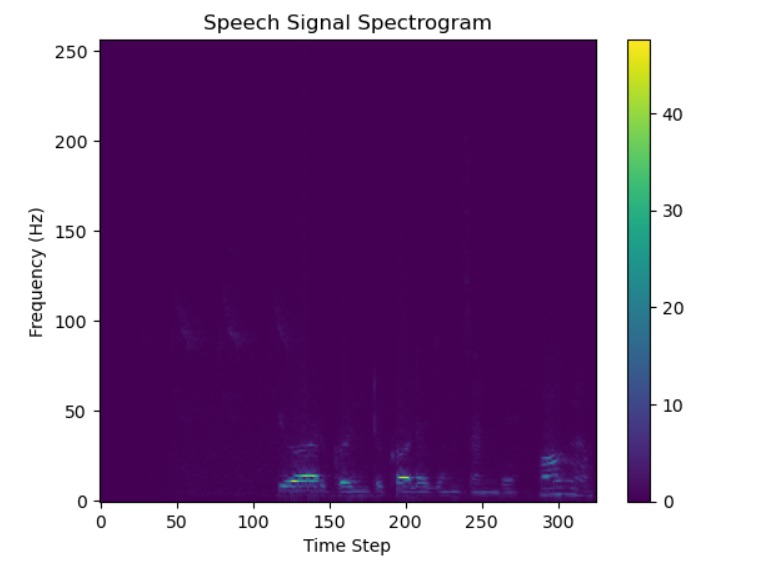
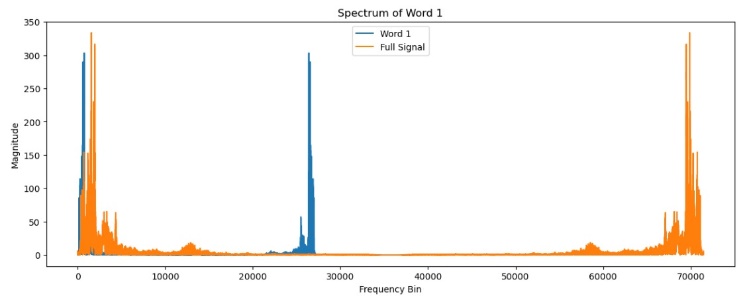
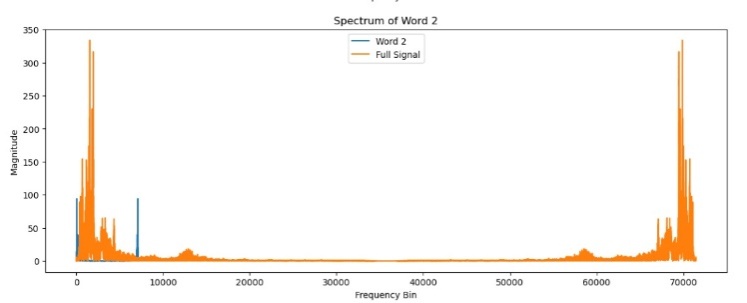
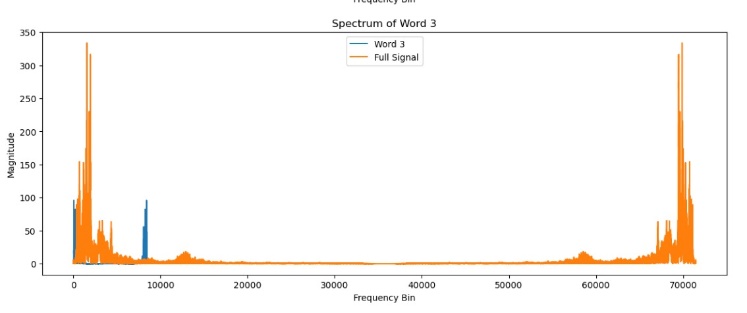
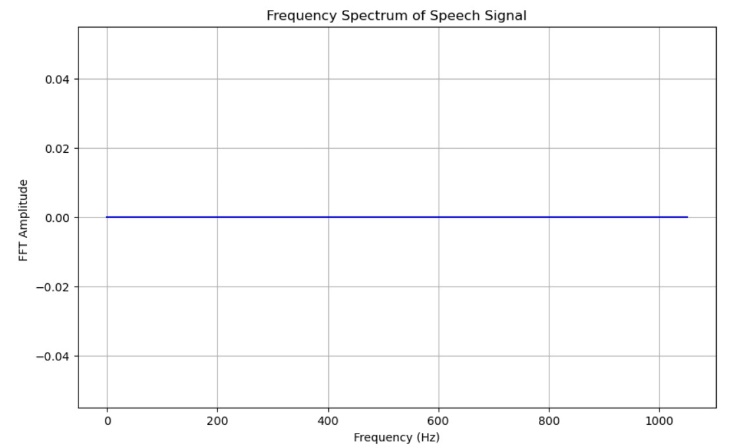
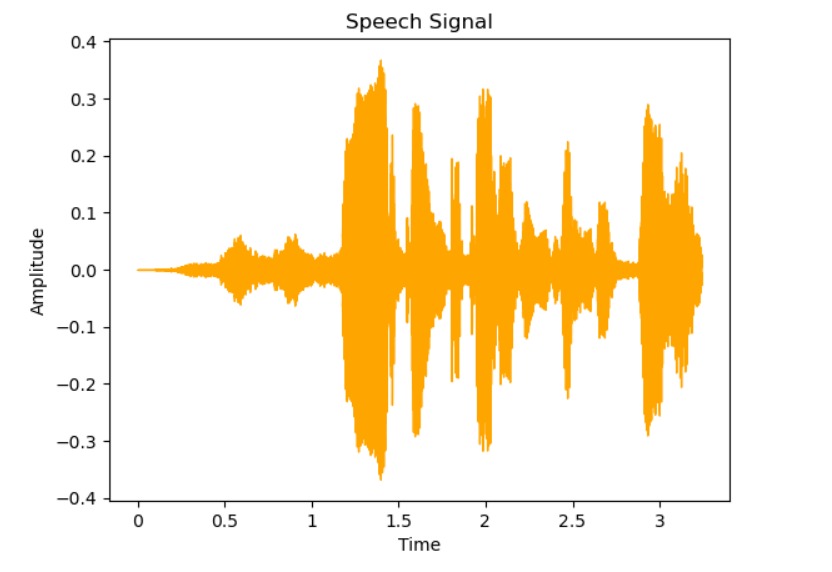
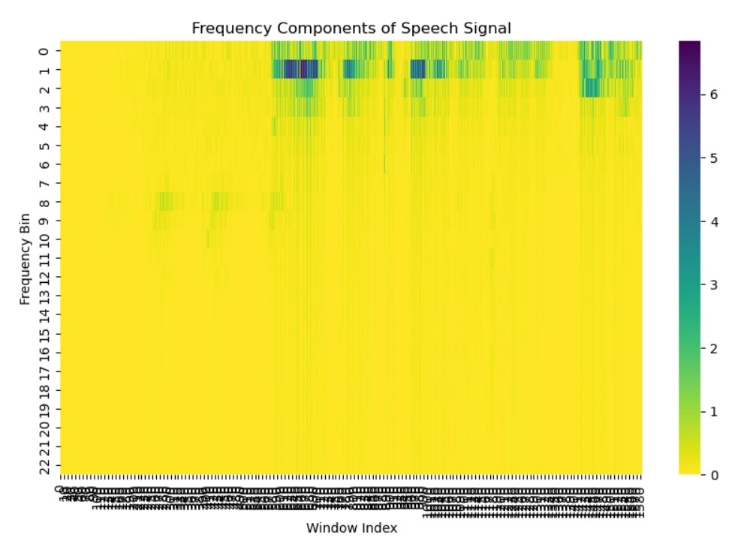
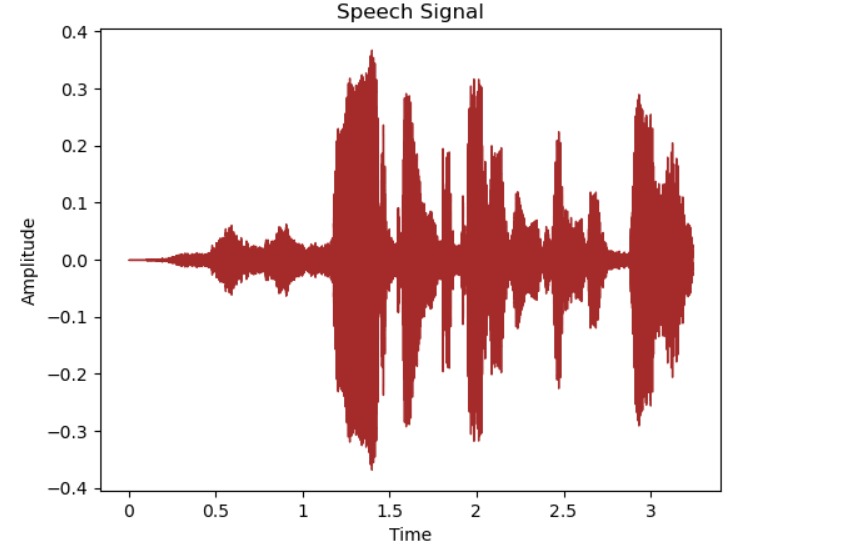
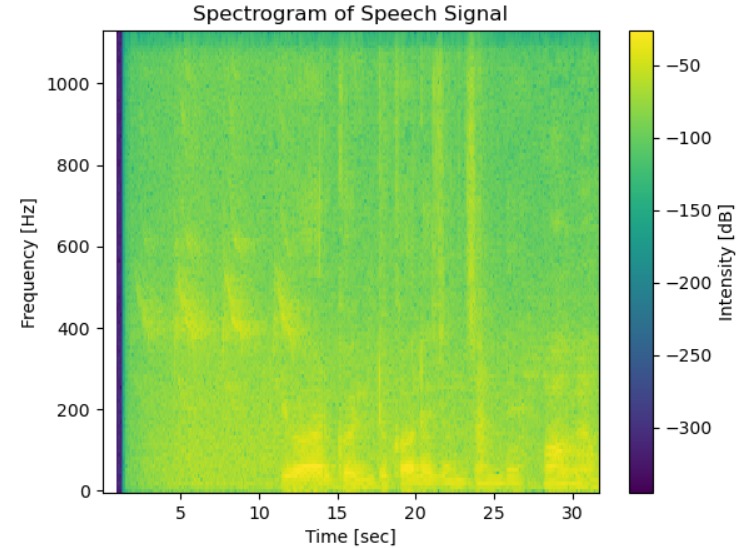
**Lab-2**

****

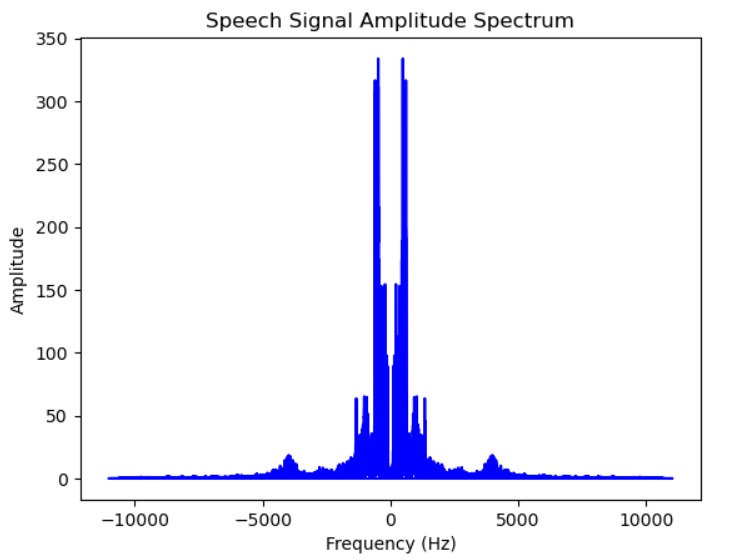
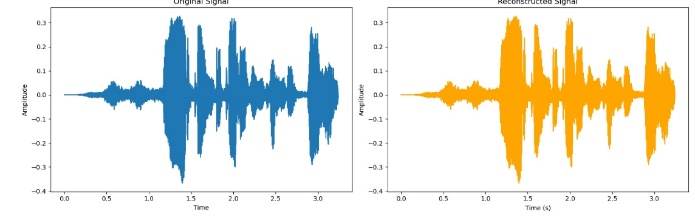
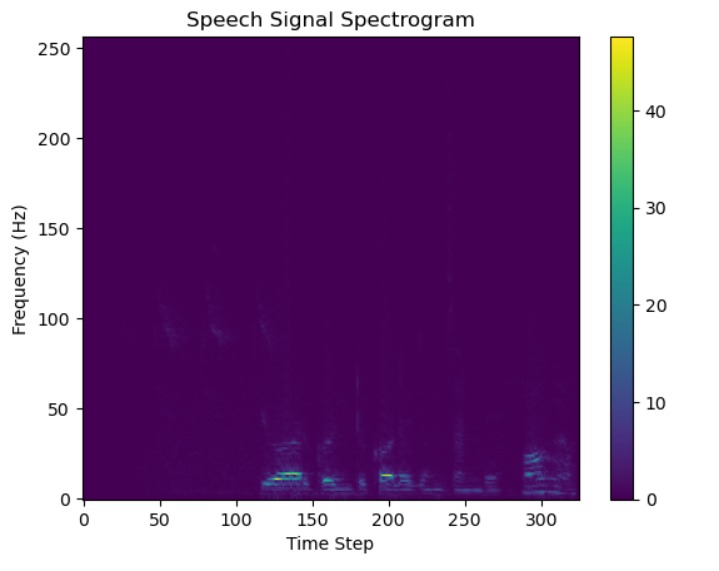
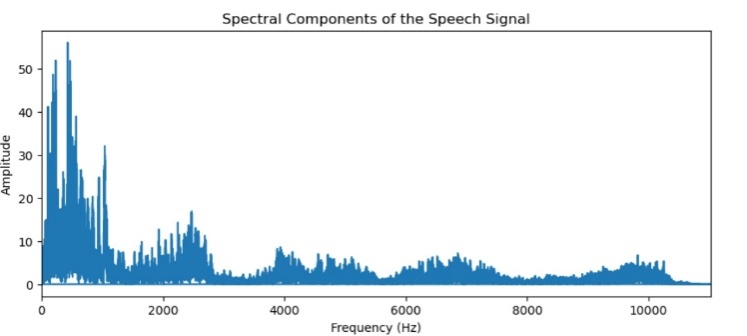
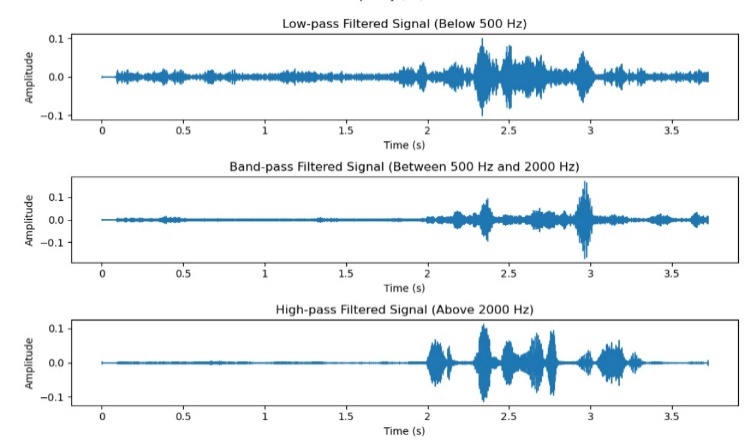
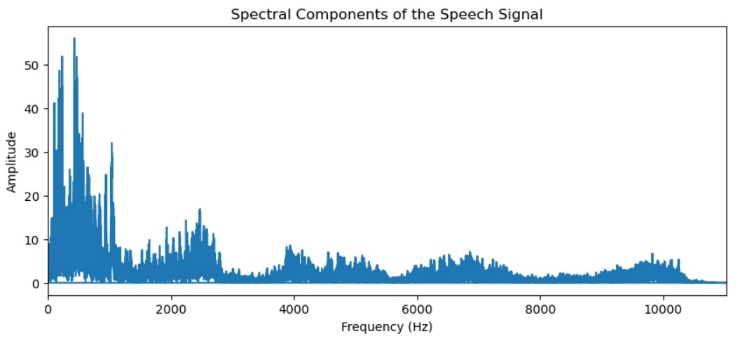
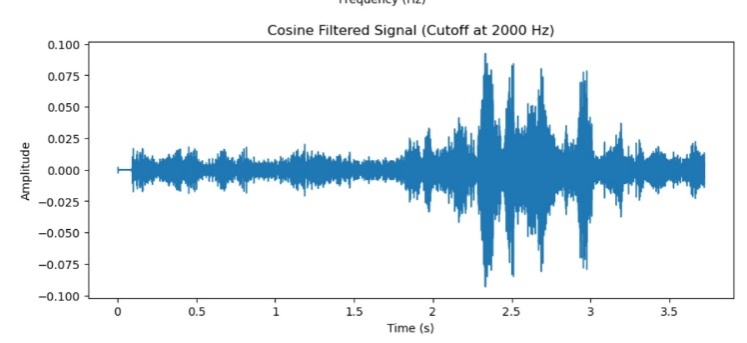
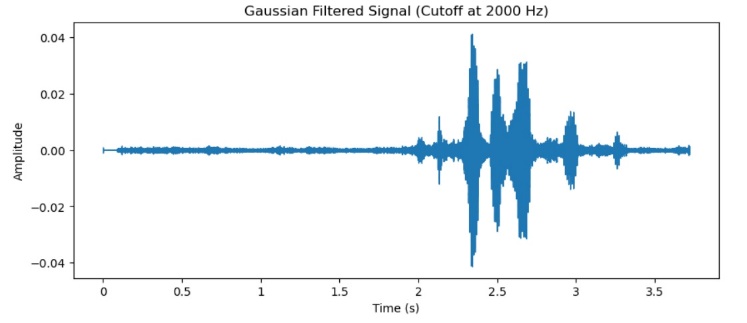
**Lab-3**

****

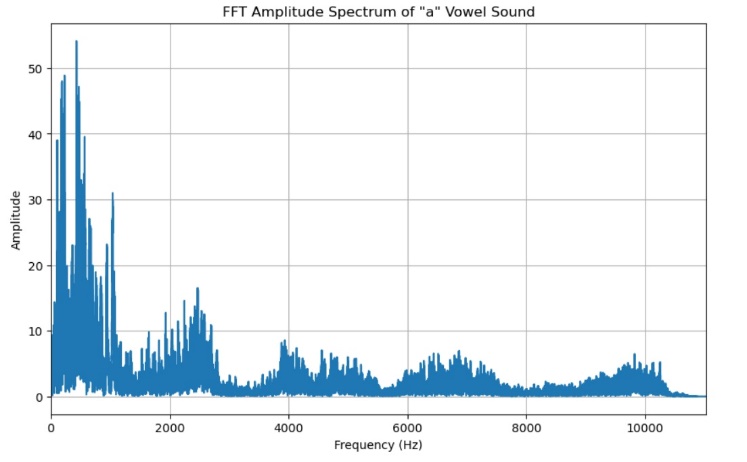
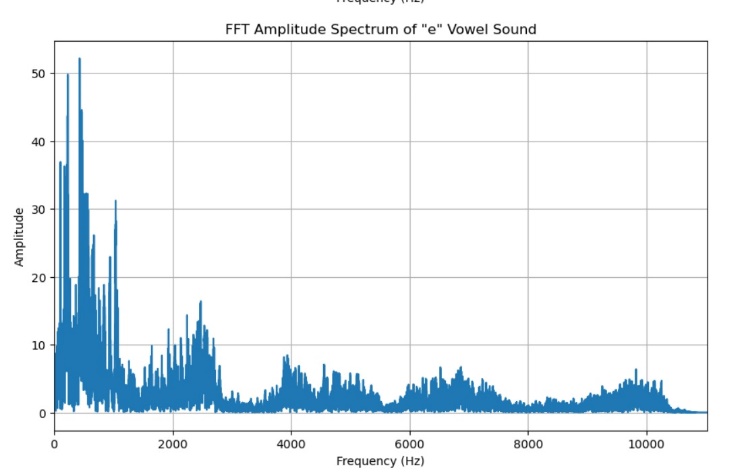
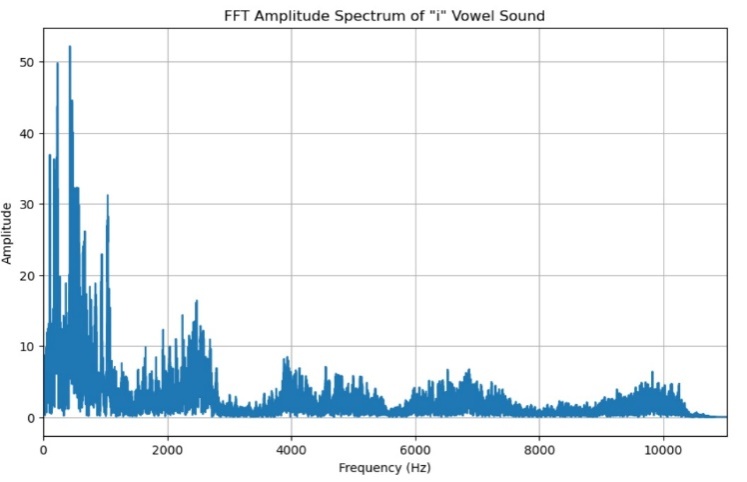
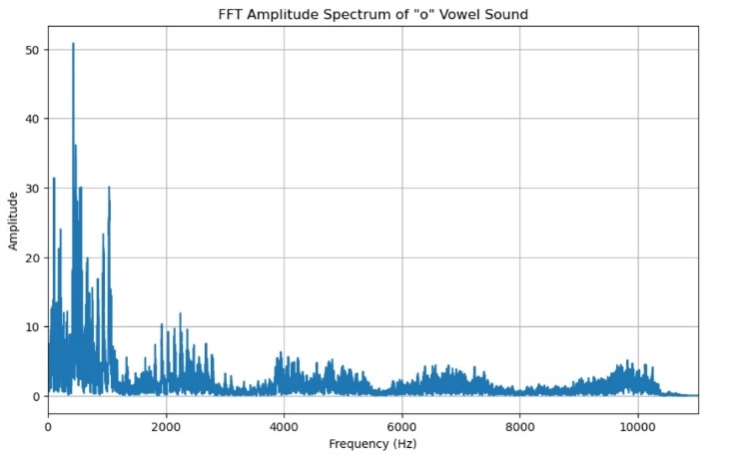
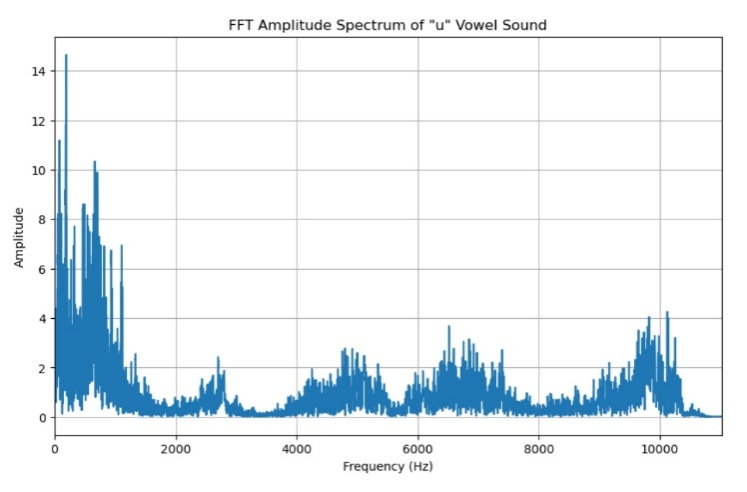
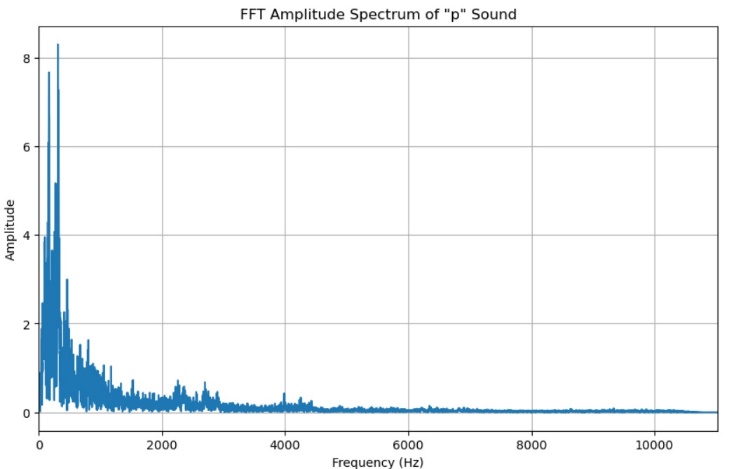
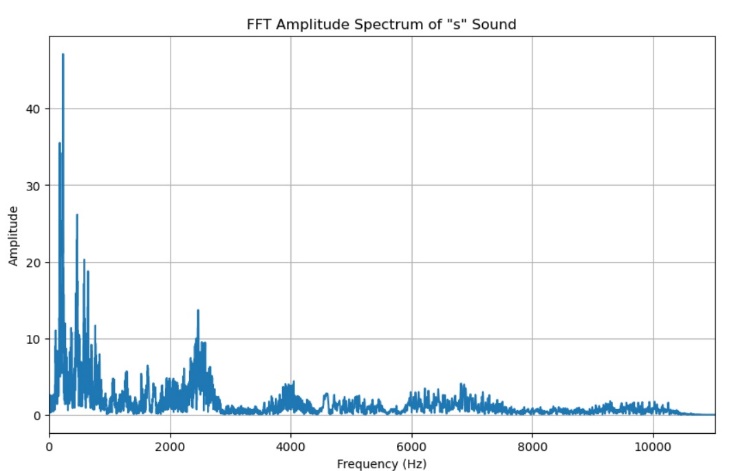
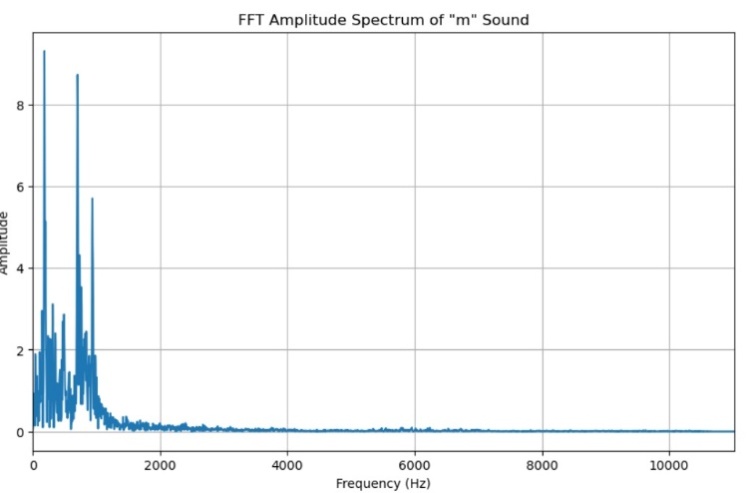
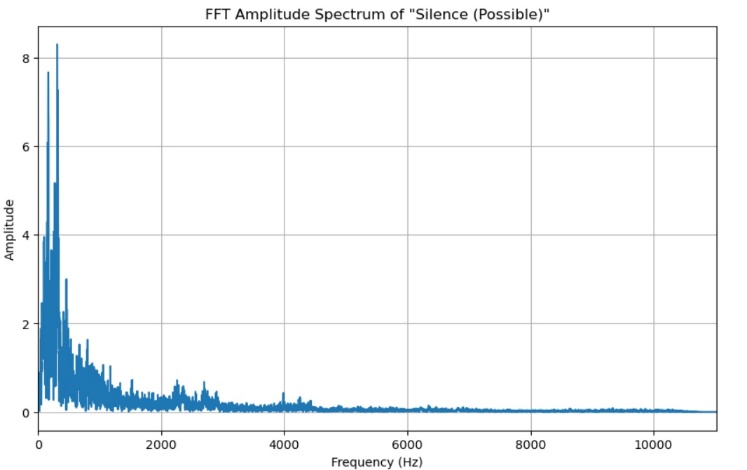
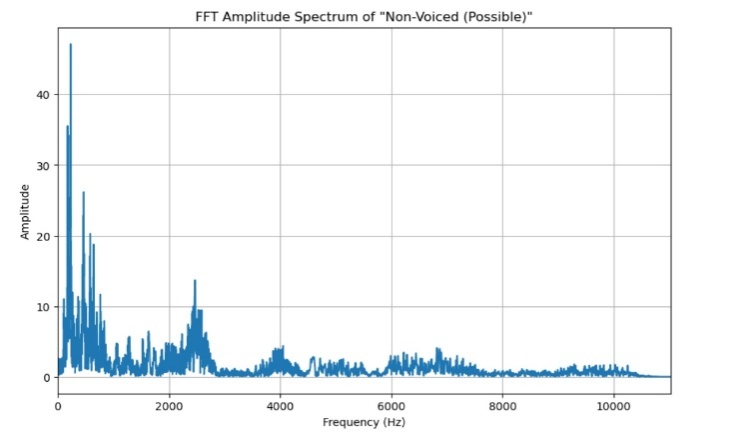
**Lab-4**

****

**Lab-5**

****

**Lab-6**

****

|  |
| --- |
| **Lab-7**  **WhatsApp Image 2024-04-08 at 14.05.24.jpegWhatsApp Image 2024-04-08 at 14.05.09.jpegWhatsApp Image 2024-04-08 at 14.04.55.jpegWhatsApp Image 2024-04-08 at 14.04.25.jpeg**  **Lab-8**  Screenshot 2024-04-13 212010.pngScreenshot 2024-04-13 212036.pngScreenshot 2024-04-13 212045.pngScreenshot 2024-04-13 212054.pngScreenshot 2024-04-13 212104.png  REFERENCES   1. Das, A., Guha, S., Singh, P.K., Ahmadian, A., Senu, N. and Sarkar, R., 2020. A hybrid meta-heuristic feature selection method for identification of Indian spoken languages from audio signals. IEEE Access, 8, pp.181432-181449. 2. Dey, S., Sahidullah, M. and Saha, G., 2022. An overview of Indian spoken language recognition from machine learning perspective. ACM Transactions on Asian and Low-Resource Language Information Processing, 21(6), pp.1-45. 3. Singh, G., Sharma, S., Kumar, V., Kaur, M., Baz, M. and Masud, M., 2021. Spoken language identification using deep learning. Computational Intelligence and Neuroscience, 2021. 4. Biswas, M., Rahaman, S., Ahmadian, A., Subari, K. and Singh, P.K., 2023. Automatic spoken language identification using MFCC based time series features. Multimedia Tools and Applications, 82(7), pp.9565-9595. 5. Garain, A., Singh, P.K. and Sarkar, R., 2021. FuzzyGCP: A deep learning architecture for automatic spoken language identification from speech signals. Expert Systems with Applications, 168, p.114416. 6. Basu, J., Khan, S., Roy, R., Basu, T.K. and Majumder, S., 2021. Multilingual speech corpus in low-resource eastern and northeastern indian languages for speaker and language identification. Circuits, Systems, and Signal Processing, 40(10), pp.4986-5013. |