Report on Spoken Language Identification Project

Objective:

The primary aim of this project is to develop a system capable of identifying spoken languages from audio data. The focus is on processing speech data, extracting meaningful features, and using these features for accurate classification of languages.

Dataset:

The dataset used for this project is the "IIIT Spoken Language Datasets" which consists of audio files categorized by language. The dataset includes the following languages:
- Tamil
- Hindi
- Bengali
- Kannada
- Telugu
- Marathi
- Malayalam

Methodology:

1. Data Preprocessing:

- Audio files are loaded and processed using the `librosa` library.
- Each audio file undergoes feature extraction to derive essential characteristics for classification.

2. Feature Extraction:

- MFCC (Mel-frequency cepstral coefficients) features are extracted from each audio sample. These features are widely used in speech and audio processing due to their ability to represent the power spectrum of audio signals in a compact form.
- A fixed number of frames (500)each consisting 20 extracted features, is ensured for all samples by either padding shorter sequences with zeros or truncating longer sequences. This step ensures uniformity across the dataset.

> pre_emphasize: Applies a pre-emphasis filter to the audio signal to enhance high frequencies.
> framing: Divides the audio signal into overlapping frames.
> hamming window: Applies a Hamming window to each frame to reduce spectral leakage.
> fft: Computes the Fast Fourier Transform (FFT) of the frames to obtain the magnitude spectrum.

mel_filter_bank : Creates a Mel filter bank to convert the power spectrum to the Mel scale.
apply_filter_bank: Applies the Mel filter bank to the power spectrum.
log_compression: Applies logarithmic compression to the Mel energy.
DCT: Computes the Discrete Cosine Transform (DCT) to obtain the MFCCs.
feat_extraction: Combines all the above functions to extract MFCC features from the audio signal.

3. Feature Scaling:

- The extracted features are normalized using StandardScaler from the sklearn library. This ensures that all features have a mean of 0 and a standard deviation of 1, which improves the performance of downstream models.

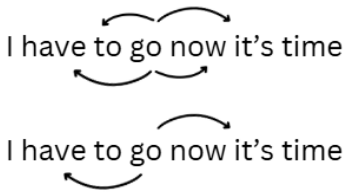
Next Steps:

Model Development

- Implement deep learning models, such as TDNN and Vanilla ANN, to classify the extracted features into respective languages.
- Alternatively, explore the use of x-vectors for feature extraction, as they are well-suited for speaker and language recognition tasks.

Model Architecture:

- Shallow Neural network, where the input layer consists of 128 neurons and final layer consists of 7 neurons.
- Three TDNN layers one takes previous 2 and future 2 values. Another layer takes n-2 and n+2 and last takes n-3 and n+3.



Both these applied with batch normalization.

Evaluation:

- Split the dataset into training, validation, and testing subsets.
- Use appropriate metrics like accuracy to evaluate model performance.