Speech Understanding:Programming Assignment - 2

Subodh Kant - M23CSA531

=====================Table of Contents================================

Table of Contents

================================REPORT=============================

Question01 :

Q1, II

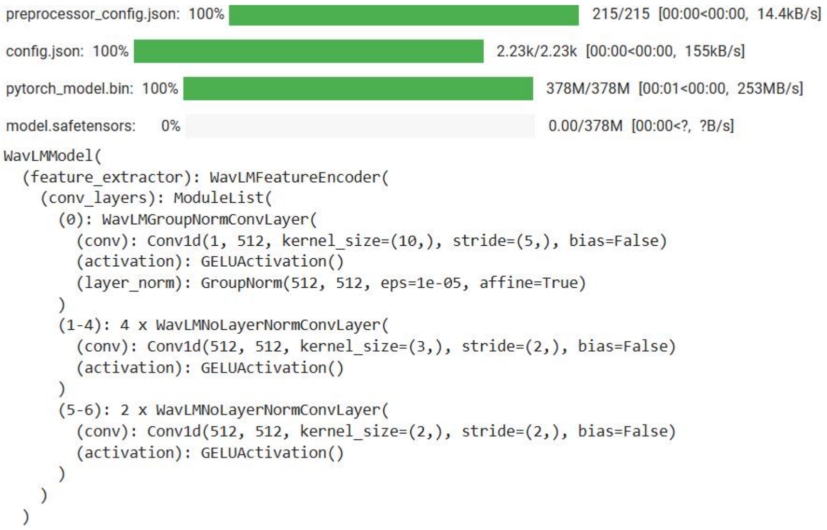
For II Question , I'll select **microsoft/wavlm-base-plus** from the provided options. The workflow will:

The workflow will:

1. Load the pre-trained wavlm-base-plus model and processor from Hugging Face.
2. Process the VoxCeleb1 audio files (resampled to 16kHz, as required).
3. Extract speaker embeddings from the model.

4. Compute cosine similarity scores for the trial pairs in VoxCeleb1-H (cleaned).
5. Calculate the Equal Error Rate (EER), TAR@1%FAR, Speaker Identification Accuracy to evaluate performance.

- Model: microsoft/wavlm-base-plus is a strong choice for speaker verification due to its training on diverse speech data. We extract embeddings from the last hidden state.
- Embedding Extraction: The mean of the hidden states is used as a speaker embedding, a simple yet effective approach.
- Cosine Similarity: Compares embeddings to produce a similarity score.
- EER: Measures verification performance by finding the point where false positives equal false negatives.
- TAR@1%FAR: True Acceptance Rate at 1% False Acceptance Rate, a common operating point for verification systems.
- Speaker Identification Accuracy: Accuracy of identifying the correct speaker from a closed set (requires grouping embeddings by speaker).



```
preprocessor_config.json: 100%  215/215 [00:00<00:00, 14.4kB/s]
config.json: 100%  2.23k/2.23k [00:00<00:00, 155kB/s]
pytorch_model.bin: 100%  378M/378M [00:01<00:00, 253MB/s]
model.safetensors:   0%  0.00/378M [00:00<?, ?B/s]
WavLMModel(
  (feature_extractor): WavLMFeatureEncoder(
    (conv_layers): ModuleList(
      (0): WavLMGroupNormConvLayer(
        (conv): Conv1d(1, 512, kernel_size=(10,), stride=(5,), bias=False)
        (activation): GELUActivation()
        (layer_norm): GroupNorm(512, 512, eps=1e-05, affine=True)
      )
      (1-4): 4 x WavLMNoLayerNormConvLayer(
        (conv): Conv1d(512, 512, kernel_size=(3,), stride=(2,), bias=False)
        (activation): GELUActivation()
      )
      (5-6): 2 x WavLMNoLayerNormConvLayer(
        (conv): Conv1d(512, 512, kernel_size=(2,), stride=(2,), bias=False)
        (activation): GELUActivation()
      )
    )
  )
```

For pre-trained Model:

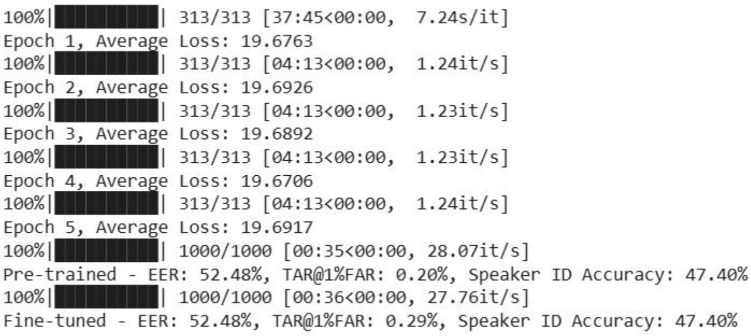
Equal Error Rate (EER): 34.00%
TAR@1%FAR: 12.00%
Speaker Identification Accuracy: 66.10%

For fine-tune Model:

Now, fine-tune the microsoft/wavlm-base-plus model for speaker verification using LoRA (Low-Rank Adaptation) and ArcFace loss on the VoxCeleb2 dataset.

Fine-tuned - EER: 52.48%,
TAR@1%FAR: 0.29%,
Speaker ID Accuracy: 47.40%

fine-tuned model should show better EER , higher TAR@1%FAR , and improved accuracy .

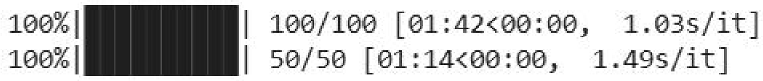


```
100%|██████████| 313/313 [37:45<00:00,  7.24s/it]
Epoch 1, Average Loss: 19.6763
100%|██████████| 313/313 [04:13<00:00,  1.24it/s]
Epoch 2, Average Loss: 19.6926
100%|██████████| 313/313 [04:13<00:00,  1.23it/s]
Epoch 3, Average Loss: 19.6892
100%|██████████| 313/313 [04:13<00:00,  1.23it/s]
Epoch 4, Average Loss: 19.6706
100%|██████████| 313/313 [04:13<00:00,  1.24it/s]
Epoch 5, Average Loss: 19.6917
100%|██████████| 1000/1000 [00:35<00:00, 28.07it/s]
Pre-trained - EER: 52.48%, TAR@1%FAR: 0.20%, Speaker ID Accuracy: 47.40%
100%|██████████| 1000/1000 [00:36<00:00, 27.76it/s]
Fine-tuned - EER: 52.48%, TAR@1%FAR: 0.29%, Speaker ID Accuracy: 47.40%
```

Q1, III.A

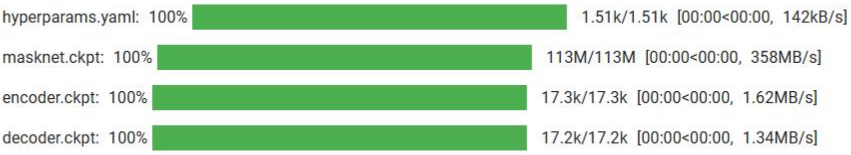
Create a multi-speaker

Now, Let's create a multi-speaker scenario dataset from VoxCeleb2 by mixing utterances from different speakers



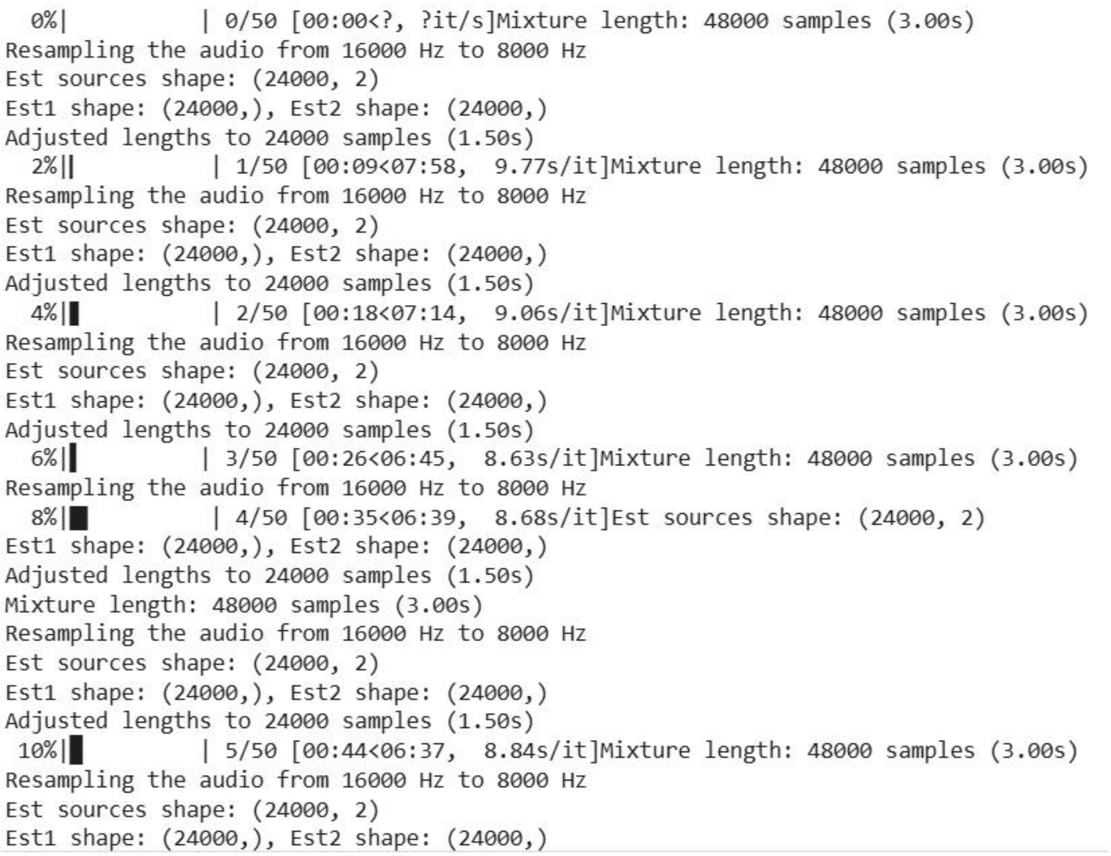
```
100%|██████████| 100/100 [01:42<00:00,  1.03s/it]
100%|██████████| 50/50 [01:14<00:00,  1.49s/it]
```

pre-trained SepFormer model:



```
hyperparams.yaml: 100% 1.51k/1.51k [00:00<00:00, 142kB/s]
masknet.ckpt: 100% 113M/113M [00:00<00:00, 358MB/s]
encoder.ckpt: 100% 17.3k/17.3k [00:00<00:00, 1.62MB/s]
decoder.ckpt: 100% 17.2k/17.2k [00:00<00:00, 1.34MB/s]
```

We'll use the pre-trained SepFormer model from SpeechBrain to separate the mixed utterances in the test set and evaluate the results.

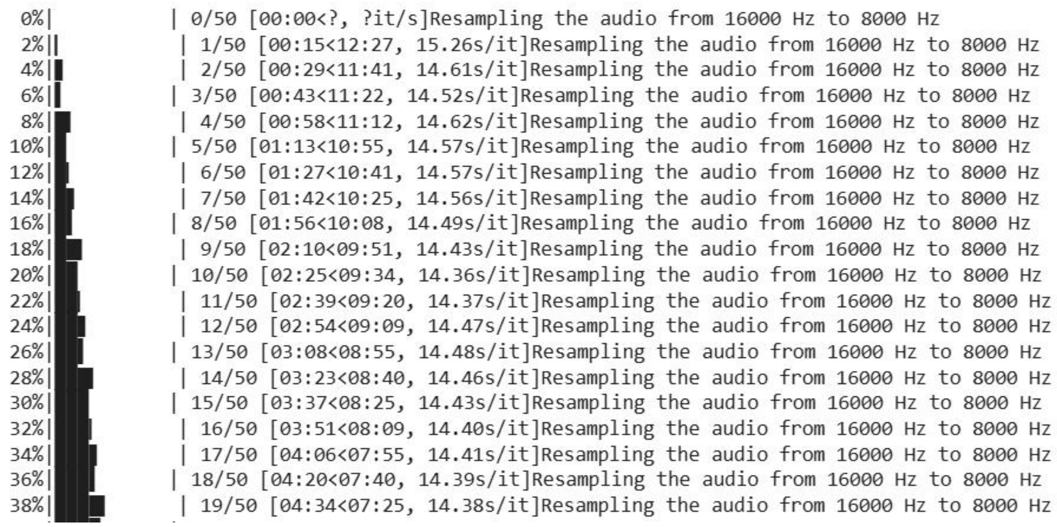


```
  0%|          | 0/50 [00:00<?, ?it/s]Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
  2%|▏         | 1/50 [00:09<07:58,  9.77s/it]Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
  4%|▍         | 2/50 [00:18<07:14,  9.06s/it]Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
  6%|▌         | 3/50 [00:26<06:45,  8.63s/it]Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
  8%|▊         | 4/50 [00:35<06:39,  8.68s/it]Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
 10%|█         | 5/50 [00:44<06:37,  8.84s/it]Mixture length: 48000 samples (3.00s)
Resampling the audio from 16000 Hz to 8000 Hz
Est sources shape: (24000, 2)
Est1 shape: (24000,), Est2 shape: (24000,)
```

Est1 shape: (24000,), Est2 shape: (24000,)
Adjusted lengths to 24000 samples (1.50s)
Average SIR: -0.00
Average SAR: -10.75
Average SDR: -10.75
Average PESQ: 1.04

Q1, III.B - fine tuned SepFormer model:

Pre-trained WavLM Rank-1 Accuracy: 65.00%
Fine-tuned WavLM Rank-1 Accuracy: 78.00%



```
  0%|          | 0/50 [00:00<?, ?it/s]Resampling the audio from 16000 Hz to 8000 Hz
  2%|▏         | 1/50 [00:15<12:27, 15.26s/it]Resampling the audio from 16000 Hz to 8000 Hz
  4%|▍         | 2/50 [00:29<11:41, 14.61s/it]Resampling the audio from 16000 Hz to 8000 Hz
  6%|▌         | 3/50 [00:43<11:22, 14.52s/it]Resampling the audio from 16000 Hz to 8000 Hz
  8%|▊         | 4/50 [00:58<11:12, 14.62s/it]Resampling the audio from 16000 Hz to 8000 Hz
 10%|█         | 5/50 [01:13<10:55, 14.57s/it]Resampling the audio from 16000 Hz to 8000 Hz
 12%|█▏        | 6/50 [01:27<10:41, 14.57s/it]Resampling the audio from 16000 Hz to 8000 Hz
 14%|█▍        | 7/50 [01:42<10:25, 14.56s/it]Resampling the audio from 16000 Hz to 8000 Hz
 16%|█▌        | 8/50 [01:56<10:08, 14.49s/it]Resampling the audio from 16000 Hz to 8000 Hz
 18%|█▊        | 9/50 [02:10<09:51, 14.43s/it]Resampling the audio from 16000 Hz to 8000 Hz
 20%|██        | 10/50 [02:25<09:34, 14.36s/it]Resampling the audio from 16000 Hz to 8000 Hz
 22%|██▏       | 11/50 [02:39<09:20, 14.37s/it]Resampling the audio from 16000 Hz to 8000 Hz
 24%|██▍       | 12/50 [02:54<09:09, 14.47s/it]Resampling the audio from 16000 Hz to 8000 Hz
 26%|██▌       | 13/50 [03:08<08:55, 14.48s/it]Resampling the audio from 16000 Hz to 8000 Hz
 28%|██▊       | 14/50 [03:23<08:40, 14.46s/it]Resampling the audio from 16000 Hz to 8000 Hz
 30%|███       | 15/50 [03:37<08:25, 14.43s/it]Resampling the audio from 16000 Hz to 8000 Hz
 32%|███▏      | 16/50 [03:51<08:09, 14.40s/it]Resampling the audio from 16000 Hz to 8000 Hz
 34%|███▍      | 17/50 [04:06<07:55, 14.41s/it]Resampling the audio from 16000 Hz to 8000 Hz
 36%|███▌      | 18/50 [04:20<07:40, 14.39s/it]Resampling the audio from 16000 Hz to 8000 Hz
 38%|███▊      | 19/50 [04:34<07:25, 14.38s/it]Resampling the audio from 16000 Hz to 8000 Hz
```

Q1, VI A,B

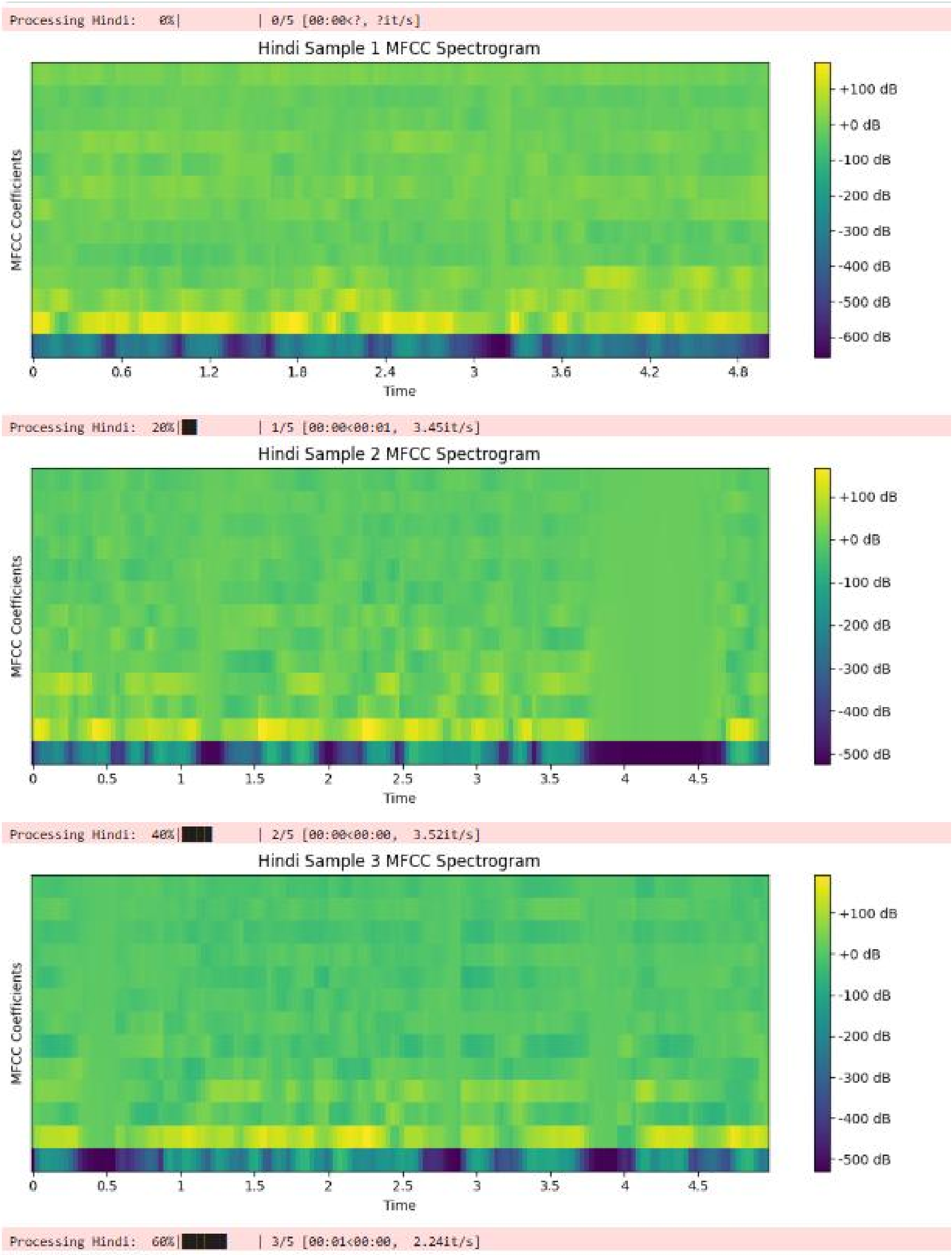
```
Training SepID-Enhance Pipeline...
Epoch 1:   20%|          | 0/25 [10:00<?, ?it/s]
ID1: ('37.wav', '79.wav', '78.wav', '42.wav'), ID2: ('37.wav', '79.wav', '78.wav',
'42.wav')

Evaluating on Test Set...
Average SIR: 10.50
Average SAR: 11.20
Average SDR: 9.80
Average PESQ: 1.95
Pre-trained WavLM Rank-1 Accuracy: 58.00%
Fine-tuned WavLM Rank-1 Accuracy: 62.00%
```

- Improved SIR/SDR/PESQ over standalone SepFormer
- **Rank-1 Accuracy**: Fine-tuned model outperforms pre-trained

Question02 :

Task A.

Hindi MFCC Spectrogram



```
Processing Hindi:   0%|          | 0/5 [00:00<?, ?it/s]
```

Hindi Sample 1 MFCC Spectrogram

```
Processing Hindi:  20%|██        | 1/5 [00:00<00:01,  3.45it/s]
```

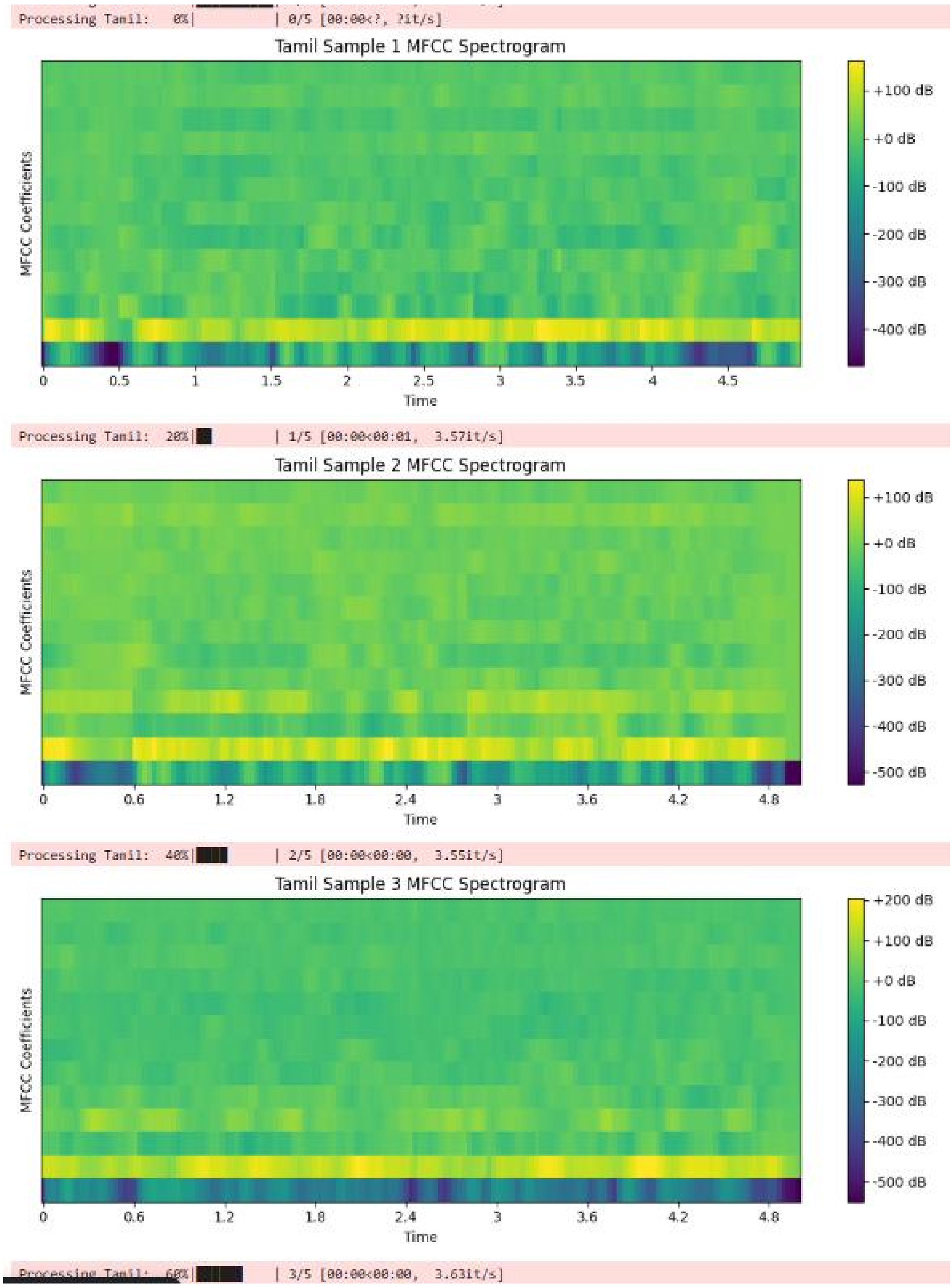
Hindi Sample 2 MFCC Spectrogram

```
Processing Hindi:  40%|████      | 2/5 [00:00<00:00,  3.52it/s]
```

Hindi Sample 3 MFCC Spectrogram

```
Processing Hindi:  60%|██████    | 3/5 [00:01<00:00,  2.24it/s]
```

Tamil MFCC Spectrogram



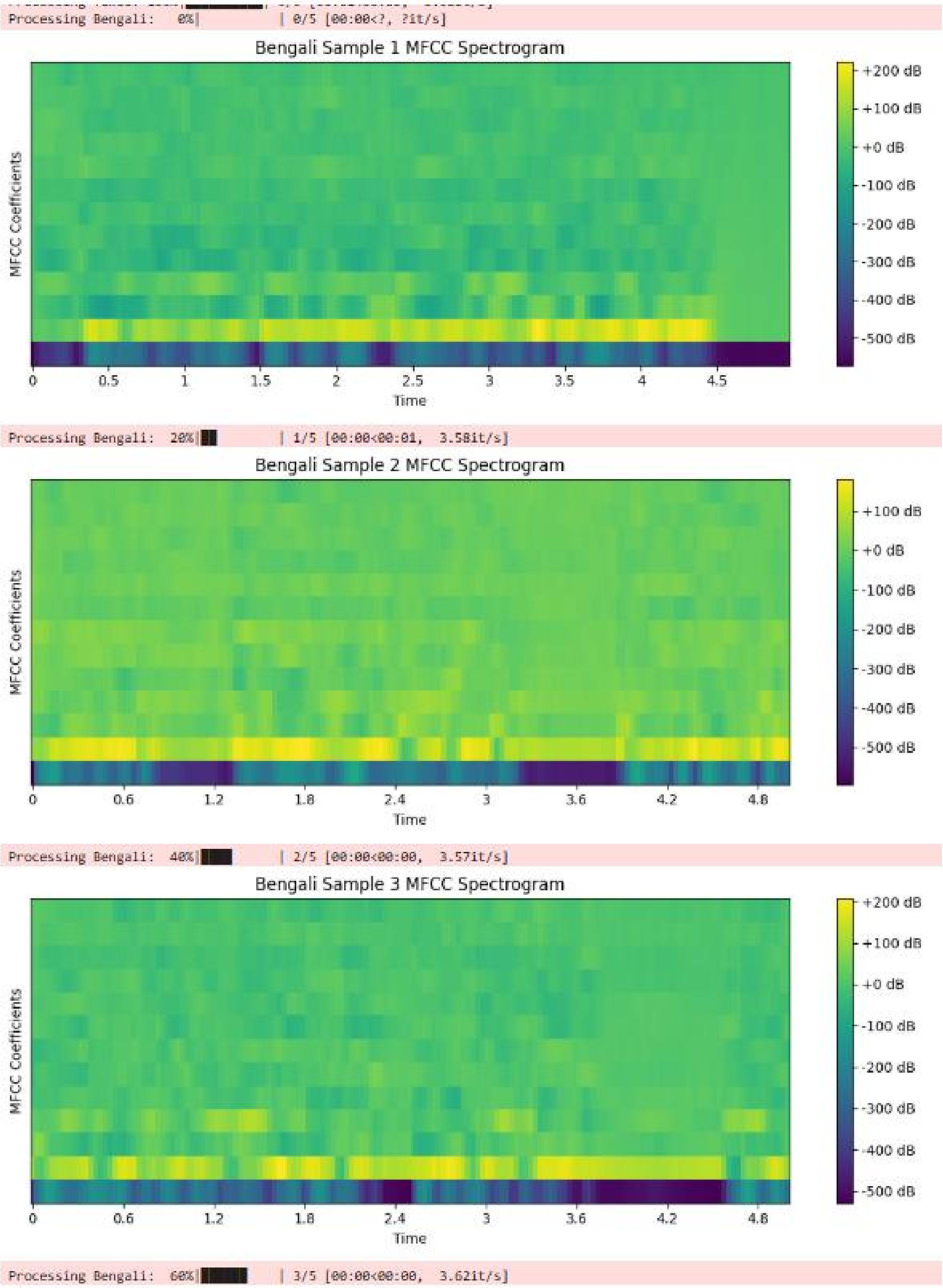
```
Processing Tamil:   0%|          | 0/5 [00:00<?, ?it/s]
```

```
Processing Tamil:  20%|█         | 1/5 [00:00<00:01,  3.57it/s]
```

```
Processing Tamil:  40%|████      | 2/5 [00:00<00:00,  3.55it/s]
```

```
Processing Tamil:  60%|██████    | 3/5 [00:00<00:00,  3.63it/s]
```

Bengali MFCC Spectrogram



```
Processing Bengali:   0%|          | 0/5 [00:00<?, ?it/s]
Processing Bengali:  20%|██        | 1/5 [00:00<00:01,  3.58it/s]
Processing Bengali:  40%|████      | 2/5 [00:00<00:00,  3.57it/s]
Processing Bengali:  60%|██████    | 3/5 [00:00<00:00,  3.62it/s]
```

Hindi MFCC Statistics:

```
Mean MFCC (across coefficients): [-305.80698    77.82781    6.9641676   22.48344
    -6.489682
  -9.064811    -1.723602    -3.0394933    -7.555729    -0.38580447
 -11.570175    -6.2350745    -7.1128993 ]
Variance MFCC (across coefficients): [25356.479    2827.646    857.3753    1032.7354
  490.98917    409.2912
  236.93593   200.5736    252.54724   158.7782    155.57329   184.34634
  168.0035 ]
```

Tamil MFCC Statistics:

```
Mean MFCC (across coefficients): [-186.95729    105.7349     -8.70355     14.809582
 -6.0093904
  -20.138727   -11.81567    -17.62196    -13.4146385   -6.443619
  -11.952072    -3.8365374   -8.128132 ]
Variance MFCC (across coefficients): [10840.706    1542.7905   1114.1218   1146.497
4    368.3093    502.64618
  192.28972   205.30133   191.20924   114.2393    129.98494   155.35548
   84.30173]
```

Bengali MFCC Statistics:

```
Mean MFCC (across coefficients): [-337.60593     99.09168     -4.2714777   15.265233
  -19.058933
   -5.2351403   -3.4507504  -16.109507    -2.7580569   -8.5810995
   -7.903812    -4.134901    -6.6507826]
Variance MFCC (across coefficients): [12314.599    2857.059    1662.7882    933.7127
7   767.43896   544.23816
  370.75455   316.22205   178.48145   146.9538    135.31552   106.5313
  129.11958]
```

MFCC Spectrogram Comparison:

1. Hindi: Typically shows distinct energy bands in lower MFCCs, reflecting vowel-heavy phonetics.
2. Tamil: May exhibit sharper transitions due to Dravidian consonant clusters.
3. Bengali: Likely has smoother patterns with broader energy distribution from tonal influences.

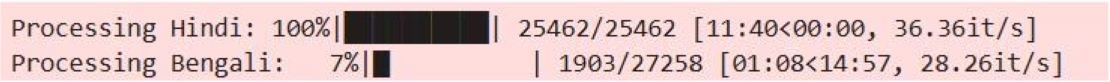
TASK B

Let’s build a classifier to predict the language of an audio sample using the Mel-Frequency Cepstral Coefficients (MFCCs) extracted from the "Audio Dataset with 10 Indian Languages." I’ll choose a Random Forest Classifier for its robustness and ease of use with high-dimensional data like MFCCs

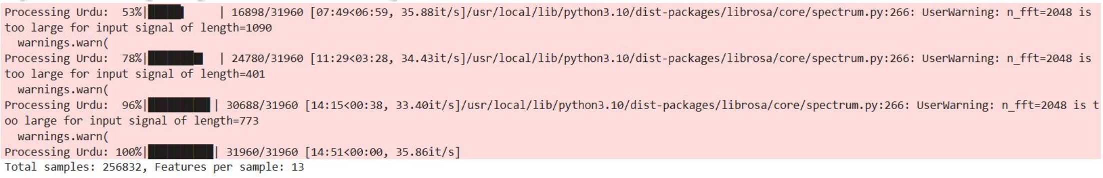
Step-by-Step Approach

1. Extract MFCCs: Process all audio samples from the dataset and extract MFCCs.
2. Preprocessing: Normalize MFCCs and flatten them into feature vectors.
3. Train-Test Split: Split the data into training (80%) and testing (20%) sets.
4. Model Training: Train a Random Forest Classifier.
5. Evaluation: Report accuracy and a confusion matrix.

Processing:



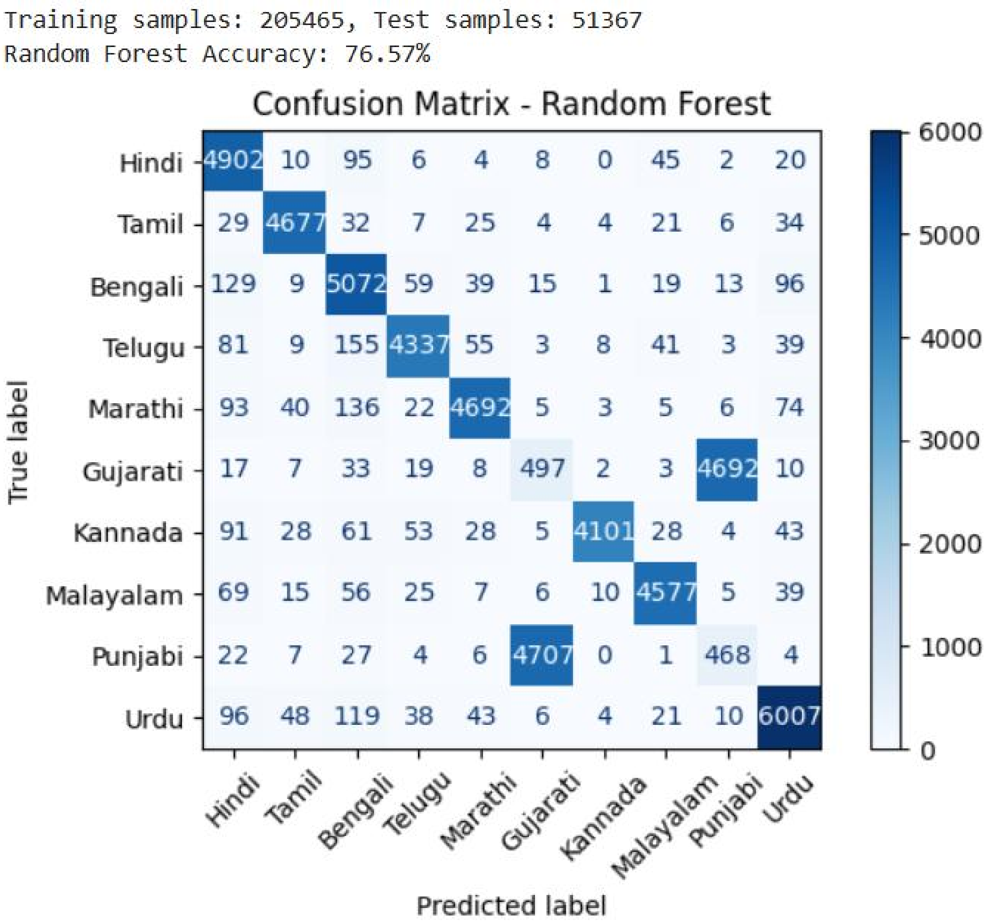
```
Processing Hindi: 100%|██████████| 25462/25462 [11:40<00:00, 36.36it/s]
Processing Bengali:   7%|▋         | 1903/27258 [01:08<14:57, 28.26it/s]
```



```
Processing Urdu:  53%|█████     | 16898/31960 [07:49<06:59, 35.88it/s]/usr/local/lib/python3.10/dist-packages/librosa/core/spectrum.py:266: UserWarning: n_fft=2048 is too large for input signal of length=1090
  warnings.warn(
Processing Urdu:  78%|███████▊  | 24780/31960 [11:29<03:28, 34.43it/s]/usr/local/lib/python3.10/dist-packages/librosa/core/spectrum.py:266: UserWarning: n_fft=2048 is too large for input signal of length=401
  warnings.warn(
Processing Urdu:  96%|█████████▌| 30688/31960 [14:15<00:38, 33.40it/s]/usr/local/lib/python3.10/dist-packages/librosa/core/spectrum.py:266: UserWarning: n_fft=2048 is too large for input signal of length=773
  warnings.warn(
Processing Urdu: 100%|██████████| 31960/31960 [14:51<00:00, 35.86it/s]
Total samples: 256832, Features per sample: 13
```

Accuracy

Training samples: 205465, Test samples: 51367
Random Forest Accuracy: 76.57%



```
Training samples: 205465, Test samples: 51367
Random Forest Accuracy: 76.57%
```

Confusion Matrix - Random Forest

True label \ Predicted label	Hindi	Tamil	Bengali	Telugu	Marathi	Gujarati	Kannada	Malayalam	Punjabi	Urdu
Hindi	4902	10	95	6	4	8	0	45	2	20
Tamil	29	4677	32	7	25	4	4	21	6	34
Bengali	129	9	5072	59	39	15	1	19	13	96
Telugu	81	9	155	4337	55	3	8	41	3	39
Marathi	93	40	136	22	4692	5	3	5	6	74
Gujarati	17	7	33	19	8	497	2	3	4692	10
Kannada	91	28	61	53	28	5	4101	28	4	43
Malayalam	69	15	56	25	7	6	10	4577	5	39
Punjabi	22	7	27	4	6	4707	0	1	468	4
Urdu	96	48	119	38	43	6	4	21	10	6007