Speech Understanding
Programming Assignment - 2

Instructions

1. All code needs to be written with Python + PyTorch only.
2. Create a repository on your Github. Push this Assignment with all of your codes, reports and readme files on that.
3. You need to submit a single zip file containing the report, codes and readme file. The zip file should be named Rollno PA2.zip.
4. In the report and the private comment of your submission on Google Classroom, you have to mention the link to your Github repository.
5. Do cite the libraries/codes/references in the report.
6. This whole assignment needs to be performed and submitted individually.

Question 1: Speech Enhancement

The task is to enhance the speech of each speaker in a multi-speaker environment.

I. Download the VoxCeleb1 dataset.

II. Take one pre-trained speaker verification model from 'hubert large', 'wav2vec2 xlsr', 'unispeech sat', and 'wavlm base plus' using this link and perform speaker verification on the VoxCeleb1-H (cleaned) dataset available at this link. Now fine-tune the selected speaker verification model using LoRA (Low-Rank Adaptation) and ArcFace loss with the list of trial pairs - VoxCeleb1 (cleaned) dataset. Compare the performance of the pre-trained and fine-tuned models on the VoxCeleb1-H (cleaned) dataset using the following metrics: EER(in %), TAR@1%FAR and Speaker Identification Accuracy.

III. Create a multi-speaker scenario dataset by mixing/overlapping utterances from 2 different speakers of the VoxCeleb1 dataset. Use **Dev A and Dev B** sets to create a multi-speaker training scenario, and use the **Test** set to create a multi-speaker testing scenario. Refer to this GitHub repository for mixing speaker utterances.

   A. Use the pre-trained SepFormer model to perform speaker separation and speech enhancement of each speaker on the created test set by analyzing these metrics: Signal to Interference Ratio (SIR), Signal to Artefacts Ratio (SAR), Signal to Distortion Ratio (SDR) and Perceptual Evaluation of Speech Quality (PESQ).

   B. Further, use the above pre-trained and finetuned speaker identification model (obtained in II) to identify which enhanced speech corresponds to which speaker after speaker separation. Report the Rank-1 identification accuracy on both models.

IV. Design a novel pipeline/algorithmic approach to combine the speaker identification model along with the SepFromer model to perform speaker separation with the speaker identification model and speech enhancement with the SepFormer model. Finetune/Train this new pipeline on the train set of created multi-speaker scenario dataset (in III) to perform speech enhancement of each speaker in the multi-speaker dataset.

A. Report the results on the test of the created multi-speaker scenario dataset (in III) using the following metrics: Signal to Interference Ratio (SIR), Signal to Artefacts Ratio (SAR), Signal to Distortion Ratio (SDR) and Perceptual Evaluation of Speech Quality (PESQ).

B. Also, report the Rank-1 identification accuracy using both pre-trained and finetuned models obtained in II to identify which enhanced speech corresponds to which speaker.

The report should contain observations and detailed analyses of the changes observed in performance throughout the experiment.

Question 2: MFCC Feature Extraction and Comparative Analysis of Indian Languages

Task A.

1. Download the audio dataset from Kaggle: Audio Dataset with 10 Indian Languages.
2. Implement a Python program to extract the Mel-Frequency Cepstral Coefficients (MFCC) from each audio sample.
3. Generate and visualize MFCC spectrograms for a representative set of samples from at least 3 languages of your choice.
4. Compare the MFCC spectrograms across the different languages. Identify and discuss any visual differences or similarities in the spectral patterns.
    a. Optionally, perform a statistical analysis (e.g., compute the mean and variance of MFCC coefficients) to quantify differences between languages.

Task B.

1. Utilize the MFCC features extracted in Task A to build a classifier that can predict the language of an audio sample.
2. Choose a suitable model (e.g., Support Vector Machine, Random Forest, or a simple Neural Network).
3. Ensure proper data preprocessing, such as normalization and a train-test split.

- Ensure your code is well-documented and modular. Submission for both tasks includes codes, languages you have used for analysis, and the report.
- Provide a detailed analysis discussing how the MFCC features reflect the acoustic characteristics of the different languages.
- Discuss potential challenges in using MFCCs to differentiate between languages, considering factors such as speaker variability, background noise, and regional accents.