ECE6255: Noise Supression Challenge

This is the repository for ECE6255 Term Project: Noise Suppression for Speech Signals.

Submitted by: Aanish Nair, Ningyuan Yang, Prithwijit Chowdhury and Sumedh Ravi.

In this repository:

- `ReadMe.pdf` which will be helpful if you are downloading the zip file and running locally.
- Folders containing code to all the methods:
  1. Spectral Subraction
  2. MCRA
  3. NSNet2
- Audio samples (both noisy and clean) to use to run examples on the scripts
- The STOI scorer to evaluate the performances
- The Project Report
- The Project Presentation
- Ouput graphs and extra scripts which might be relevant.

1. Spectral Subtraction

Code referenced from : https://github.com/Gauri-Prajapati/Speech_Enhancement

Step1: Setup Directory

Make sure that you have the `main.m`, `spectruesub.m` and `stationary_noise_evaluate.m` files in the same directory.

`main.m` -> Used to run the algorithm

`spectruesub.m` -> The spectral subtraction function

`stationary_noise_evaluate.m` -> Calculate the noise power spectral density

Open `MATLAB` and make sure that you are in a working directory that has both the files in the same place.

Step2: Running the script

Make sure you change the file paths to the clean and noisy speech files that you intend to use.

You can also include the path to where you want the reconstructed speech to be saved.

Run the script `main.m` by clicking the 'Run' button or by typing in 'main' in the command window.

2. MCRA

Code referenced from : https://github.com/Gauri-Prajapati/Speech_Enhancement

How to use codes in our folder

`Speech_Enhancement.m` : To perform noise suppression.

`\algorithms\improved_mcra_est.m` : The MRCA algorithm.

`add_noise.m` : To generate noisy speech audio files with different kinds of noise and different SNR.

`upsampling.ipynb` : To change the sampling rate of signals.

`plot_wave.m` : To plot waves of signals.

`plot_gain.m` : To generate plots of STOI gain in the report.

3. NSNet2

Step1: Checkpoint download

Download the .onnx checkpoint from the `NSNet2/check-point` directory and store it in your `$CHECKPOINT DIRECTORY$` or if you're running locally add the path `NSNet2/check-point/nsnet2-20ms-baseline.onnx` in place of `$CHECKPOINT DIRECTORY$` in the code.

Step2: Running the Baseline

Install `requirements`

```
pip install -r requirements.txt
```

Replace `$CHECKPOINT DIRECTORY$` , `$NOISY .WAV FILE$` and `$OUTPUT DIRECTORY$` in `run.py` with the location of your model checkpoint, noisy `.wav` file and output folder to store the filter audio file respectively.

Run

```
python3 run.py
```

Alternative

Run the `run.ipynb` file on Google Colabs if you don't have local resources. Make sure to upload the `.onnx` checkpoint to drive and mount

The code for NSNet2 is forked from Microsofts repository for the DNS Challenge.

Audio Files

The clean and the noisy audio files are stored in `audio_samples`

STOI score

We used Short-Time Objective Intelligibility score (STOI) score as the evaluation metric to compare performance among different techniques. STOI denotes a correlation of short-time temporal envelopes between clean and separated speech, and has been shown to be positively correlated to human speech intelligibility score.

Calculate your own STOI scores:

Once you have generated your filtered signal from your noise `.wav` file you can calculate your STOI score by passing your `filtered signal` and `clean` signal into the `STOI_scorer.py` file

```
 clean, fs = sf.read('$CLEAN_WAV_FILE$')
denoised, fs = sf.read('$FILTERED_WAV_FILE$')
```