SAINYA RANAKSHETRAM 2.0 SOLUTION

Sujay Kumar
Computer Science Engineering with DataScience Specialization, VIT-Vellore
suzzay19@gmail.com

30th December 2022

Abstract

The solution solely focusses on the Problem Statement 1 of the Artificial Intelligence. As per the problem statement suggests the audio files which are mixed with noise (helicopter audio + background sounds). I have produced a solution which can detect the noise by using Analog and Digital Signal Processing and then we compare it to the transcriptions by SpeechRecognition Model provided by the Carnegie-Mellon-University.

1 Processing Steps

1. I took the audio files which are provided as a dataset
2. I have converted the files to .wav format inorder for processing and visualization.
3. I manually selected the noise from the dataset which u have provided like some audio files doesn't have the audio and which are very small.
4. I manually extracted the noise and saved it in a noise folder.
5. Using Denoising Techniques I took out the noise from the audio.
6. I used a High Pass filter from the denoised audio to take out the noise.
7. I increased the sound of each audio file so that every word is legible.
8. I created a dataset which comprises all the digital information of audio.

2 Denoising Process

There are 4 main Denoising process in ADSP (Analog and Digital Signal Processing)

1. Spectral Subraction
2. Weiner Filter
3. Spectral Gating
4. Deep Learning Models

I have used Spectral Subraction which uses Fast-Fourier-Transform and Short-Time-Fourier-Transform algorithms to remove the noise which I have taken from the noise files.The power spectral subtraction method itself needs some inspection. This is a solution to a real problem which arises when we have a bad noise estimate– that magnitudes need to remain non-negative. This contributes to the presence of musical noise peaks.

3 After Denoising Process

We saw that there are multiple methods of denoising but I chose Spectral Subraction because it gives us the best results for the given dataset and the best in the audio denoising market is wav2vec by Facebook I tried installing but the complexity of GPU resources requirements are very high so I couldn't install it.

We get a output after the denoising process and I named it as int_output.mp3 so u can check it.

After the denoising process we use a low-pass filter (A low-pass filter (LPF) attenuates content above a cutoff frequency, allowing lower frequencies to pass through the filter.)

Before Passing it onto the filter we use a internal noisereduction library to take out the anamalies and noise from the audio to get a clear understanding of the voice.

4 Feature Extraction Process

After denoising the Audio we process those audio files for feature extraction to extract features from the current audio to change that particular audio.

These are the Audio Parameters which I took and saved into a "data.csv" file.

1. channels of the Audio
2. Frame Rates of the Audio
3. Sample Width (For Sampling process if needed)
4. Maximum Sound (Amplification Process)
5. Length of the Audio in ms (milliseconds)

5 Normalization and Channel Separartion of Audio

After Extraction of features in the dataset by analyzing the features I decided to manipulate the audio files using this.

1. The First Manipulation is we can see that there are 2 channels in many audio files so we separate it before giving it to the speech recognition software.
2. The "pydub" library has a method known as effects.normalize() to normalize the audio file, I also did that.
3. Then we make the audio files as staticlouder which drops the first 1 second in all the processed files and make it louder to make it legible.

6 Detection of Pauses

After all these Process we find the points which has no idea and cut those sounds as "[pause]" This gives us only the Audio which is needed to be heard.

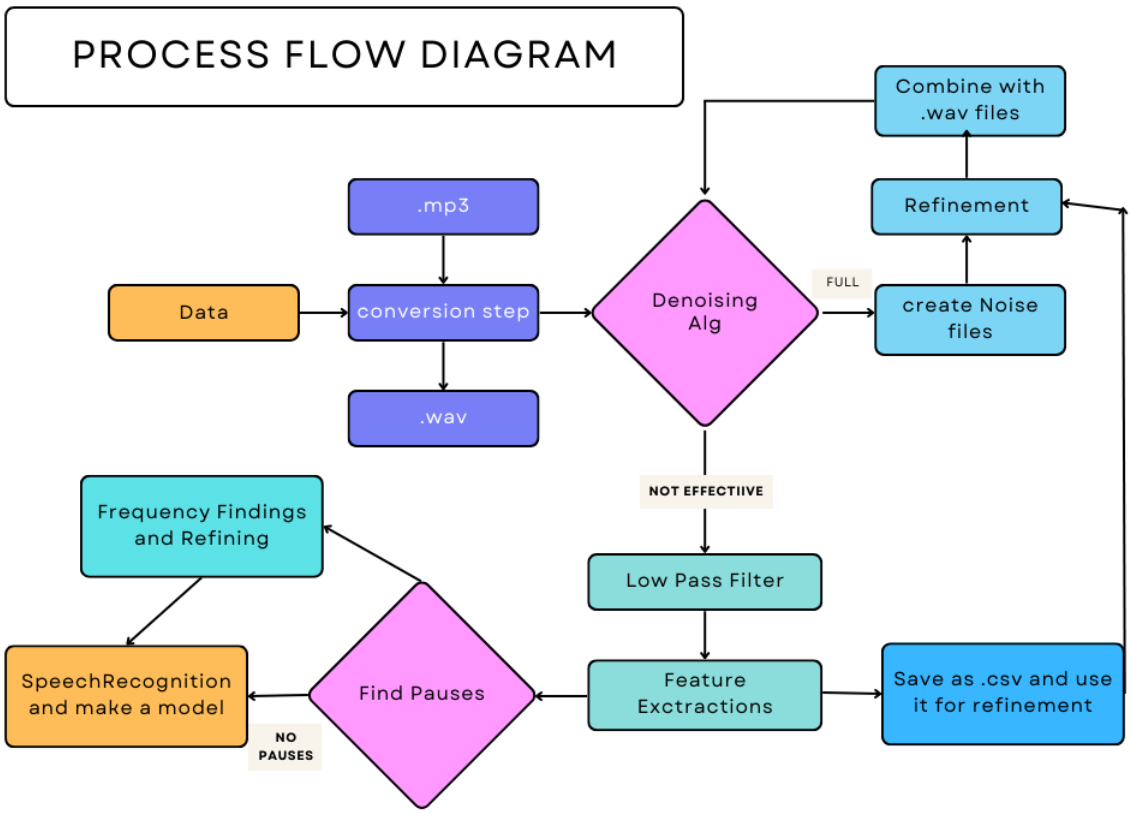
1. I found the a range of frequencies at which the pause is happening or the voice of the speaker is not audible by using the Visualization Techniques provided.
2. After finding the frequencies I found the timestamps using the librosa library and Pydub library to cut the pauses for speechrecognition module.

7 Speech Recognition

After the detection of pauses we then come to the Speech Recognition Model Now in the industry we have many speechRecognition Software but the best one to detect Hindi and English is the Google-SpeechRecognition Model but we have constraints only 50 and less than 1 hr of audio I tried using it for some of the audios and it gives the best results but I couldn't use it because of exceeded API calls and Network Error I have to come back to an offline model to use it which is CMU pocketsphinx.

"pocketsphinx" is a very good model for using the speechRecognition but unfortunately for our dataset (IndianEnglish+Hindi) The Predicted results are not very good so, making a model out of that one doesn't gives us useful results.

8 Process Flow Diagrams



9 Results and Future Works

The Results are validated and Processed in github and Drive.

The link for the result is here ...
https://drive.google.com/drive/folders/1ZRF66qHwZ5-eX2PyUI0nHGt0Wm8v0qc1?usp=share_link

The Link for the Github is here ...
https://github.com/sujaykumarmag/SR2.0

The Future works are considered within the boundaries of detecting the speechrecognition models by using Indian English by nptel dataset which is already processed by alphacephei we can use it to get amazing results.