

***A Mini-Project Report On***

**“Audio Source Separation”**

***Submitted By***

**Deepak Pawade**

**PRN: 1132210541**

**Hrishikesh Kulkarni**

**PRN: 1132210520**

**Omkar Salunkhe**

**PRN: 1132210575**

**F.Y. M.Sc. (Data Science and Big Data Analytics)**

**School of Computer Science**

**Faculty of Science**

**MIT – World Peace University**

**Pune - 411038**

**Academic Year 2021-2022**

**May - 2022**

**Dr. Vishwanath Karad MIT WORLD PEACE UNIVERSITY, PUNE**

**SCHOOL OF COMPUTER SCIENCE**

**Certificate**

This is to certify that

**Deepak Pawade**

**1132210541**

Of **M.Sc. (Data Science and Big Data Analytics)** successfully completed his/her

Mini-Project in

**“Audio Source Separation”**

to our satisfaction and submitted the same during the academic year 2021 - 2022 towards the partial fulfillment of degree of  **Master of Science in Data Science and Big Data Analytics** of Dr Vishwanath Karad MIT World Peace University under the School of Computer Science, MIT WPU, Pune.

**Prof. Dr. Shubhalaxmi Joshi Prof. Surabhi Thatte Project Mentor Name**

**Associate Dean Program Head Assistant Professor**

**Faculty of Science School of Computer School of Computer**

**Science Science**

**MITWPU MIT WPU MIT WPU**

**ACKNOWLEDGEMENT**

In the accomplishment of this project, I would like to express my special thanks of gratitude to my teachers **Prof. Project Mentor**, School of Computer Science, Dr. Vishwanath Karad MIT World Peace University whose valuable guidance has been the ones that helped me patch this project and make it full proof success. His/Her suggestions and instructions have served as the major contributor towards the completion of the project.

As we were working in a group, I would like to thank my group members for their fabulous support throughout the completion of the project. We learned a lot of things during this period, as it was hard to work in this time of adversity; we were in touch with each other throughout the period and shared everything which was important from the aspect of our project. As this project was completed by staying at home, I would also like to thank our families for their cooperation and for providing facilities to us.

**Deepak Pawade**

**1132210541**

**Contents**

| **Introduction** |  |
| --- | --- |
| Domain Name | 4 |
| Motivation | 4 |
| Problem Statement | 4 |
|  |  |
| **Literature Survey** | 5 |
|  |  |
| **Solution Design** |  |
| Solution Approach | 7 |
| Technology Stack | 7 |
| Design Model | 8 |
|  |  |
| **Solution Implementation and Results** |  |
| Obtaining Data | 11 |
| EDA | 13 |
| Pre-Processing | 14 |
| Algorithms Used | 16 |
| Results | 20 |
|  |  |
| **Conclusion and Future Work** |  |
| Conclusion | 21 |
| Future Work | 21 |
|  |  |
| **References** | 22 |

Audio Source Separation

Audio source separation is the process of extracting individual sound sources (e. g., a single flute) from a mixture of sounds (e. g., a recording of a concert band using a single microphone). Effective source separation would allow application of editing and remixing techniques to existing recordings with multiple instruments on a single track.

Separation of competing speech is a key challenge in signal processing and a feat routinely performed by the human auditory brain. A long standing benchmark of the spectrogram approach to source separation is known as the ideal binary mask. Here, we train a convolutional deep neural network, on a two- speaker cocktail party problem, to make probabilistic predictions about binary masks. Our results approach ideal binary mask performance, illustrating that relatively simple deep neural networks are capable of robust binary mask prediction.

Another application of such implementation can be in separation of interfering audio such as noise from a main sound source such as speech.

Audio Interference Separation can be one of the easier audio source separations to implement as compared to those regarding human sounds. Hence, we have first tried out implementing interference separation and then tried out a simple Multi-Layered Perceptron layer for audio source separation.

**`1.1 Problem Statement**

Implement Audio Source separation on a single channel audio.

This includes the following:

1. Audio Interference Separation.
2. Human Speech Source Separation.

**1.2 Literature Survey**

For source materials we have viewed various research papers and videos to gain insights. For audio processing, knowledge of Digital Signal Processing was required to a certain extent.

**2. Solution Design**

**2.1 Solution Approach** :

**Interfering Audio** :

Using Deep Neural Networks, an Ideal Ratio Mask can be obtained, which can help separate known interfering sounds from the required audio.

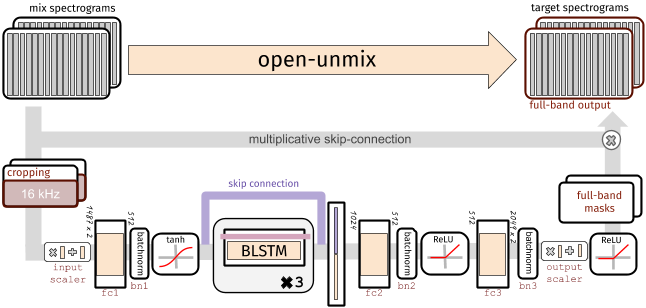
**Speech Separation**:

Once again, Deep Neural Networks can be used to create an Ideal Binary Mask which can be applied to the audio mixture for the desired separated outputs.

**2.2 Technology Stack:**

* Windows Operating System. (Linux Preferable)
* Python : For the implementation.
* VSCode : IDE
* PyTorchAudio and Librosa : Audio Processing
* PyTorch : Tensor computing with strong acceleration via GPU
* SKLearn and Tensorflow :
* Open-Unmix : a deep neural network reference implementation for music source separation

**2.3 Design Model:**

**Interfering Audio :  
 **

### **Input Stage** The input spectrogram is *standardized* using the global mean and standard deviation for every frequency bin across all frames.

### **Dimensionality reduction** In the first step after the normalization, the network learns to compress the frequency and channel axis of the model to reduce redundancy and make the model converge faster.

**Bidirectional LSTM**

The core of open-unmix is a three layer bidirectional [LSTM network](https://dl.acm.org/citation.cfm?id=1246450). Due to its recurrent nature, the model can be trained and evaluated on arbitrary length of audio signals. Since the model takes information from the past and future simultaneously, the model cannot be used in an online/real-time manner.

### **Output Stage**

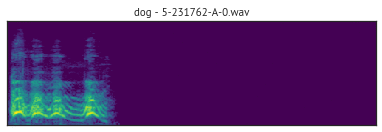
After applying the LSTM, the signal is decoded back to its original input dimensionality. In the last steps the output is multiplied with the input magnitude spectrogram, so that the model is asked to learn a mask.

**Podcast Audio**

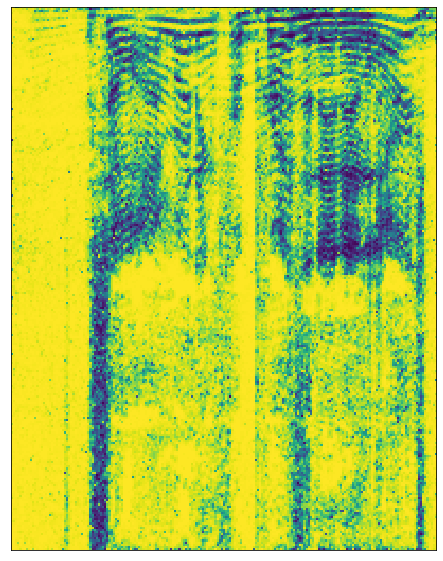
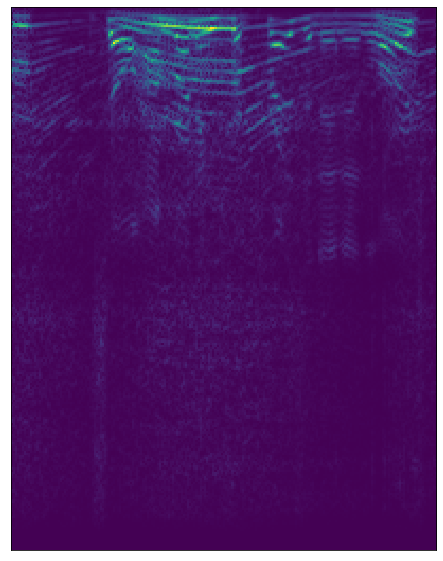
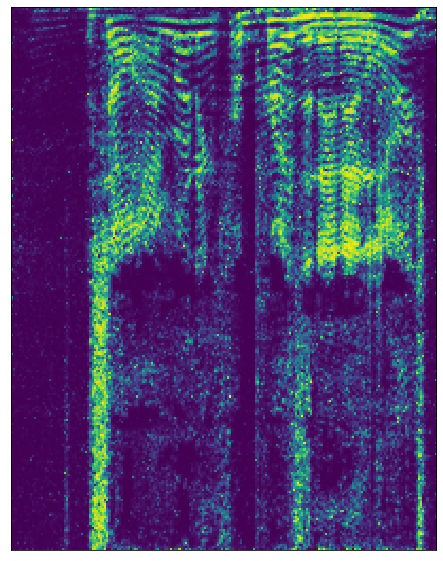
****

**+**

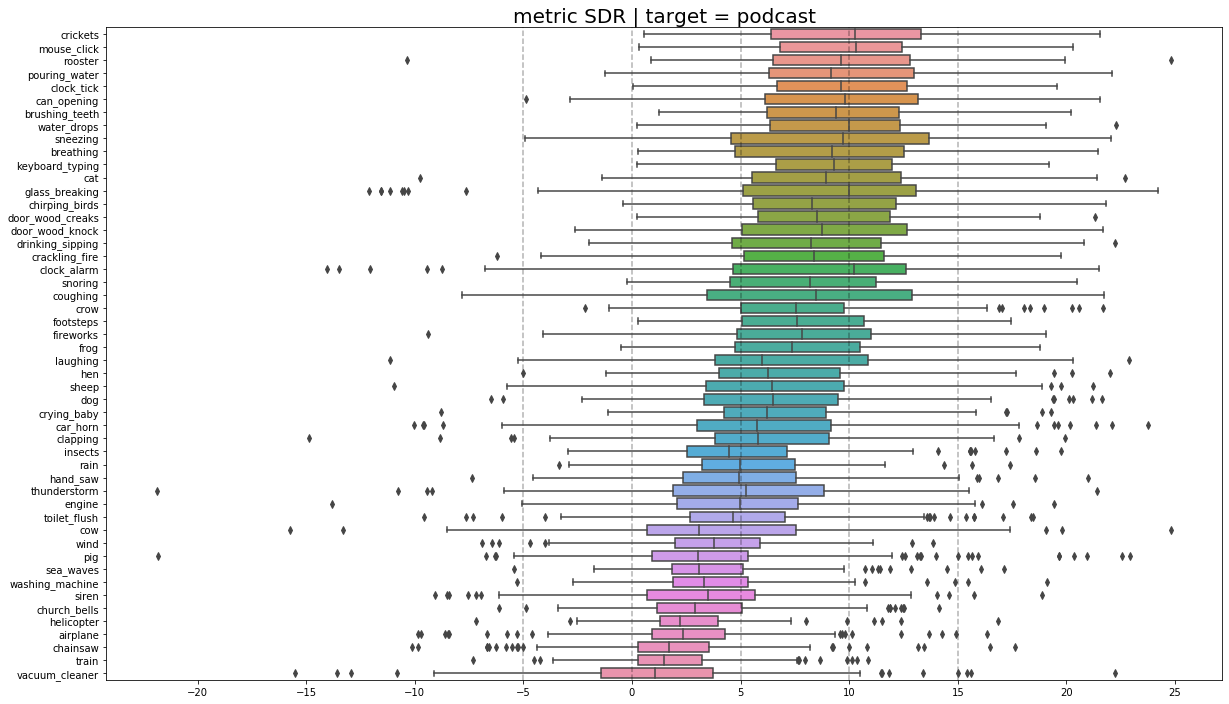
**Interfering Audio**



Ratio Mask Sources

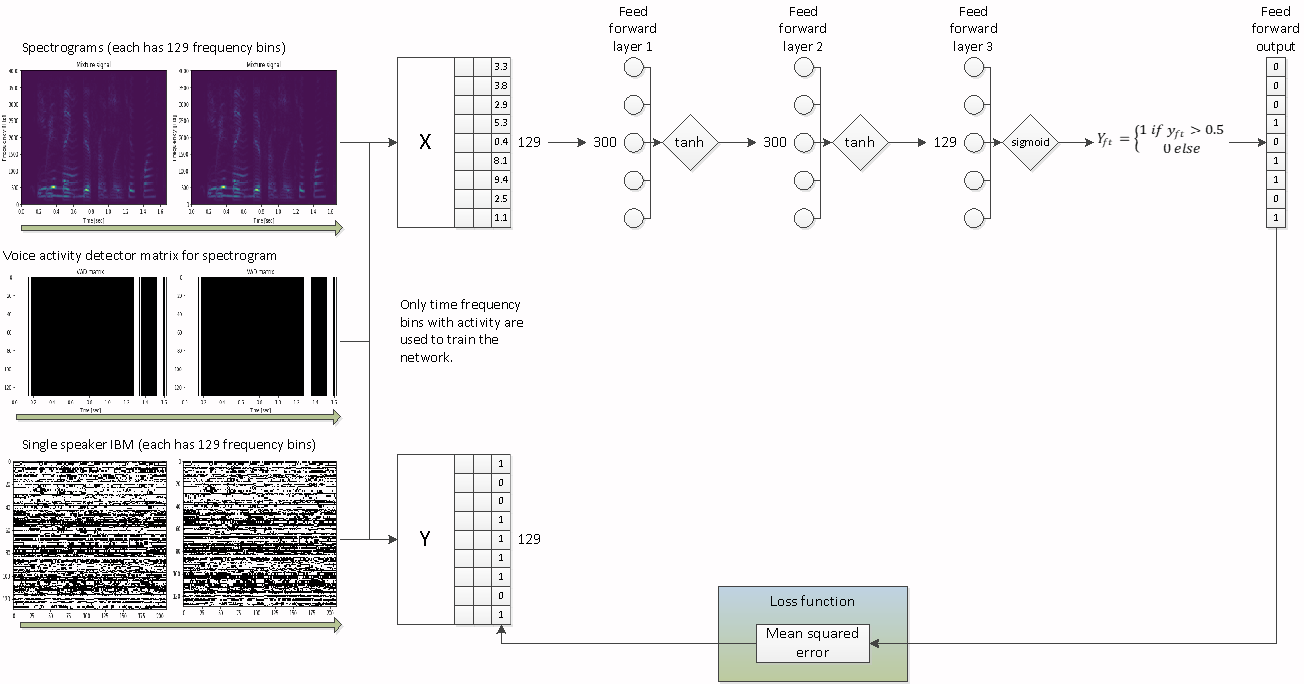


### **X = mixture**



**The Above Box plots show the SDR of interfering audios used in the creation of mixed audio sources. While most of the noises are well identifiable at certain frequencies some of them have a wide spread, hence these will be harder to separate.**

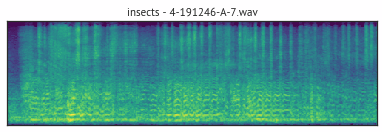
**2. Speech Separation :**

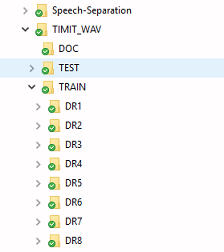
****

**Speech Separation using Feed Forward Network.**

1. **Solution Implementation and Results**

**3.1 Obtaining Data :**

1. **Interference Separation :**We collected 10 Podcasts from archive.org (JRE Podcasts). The podcasts were divided into 5sec sound files and mixed with interfering sounds from ESC 50 dataset (<https://github.com/karolpiczak/ESC-50>). The resultant mixture has characteristics such as nfft = 512, nhop = 160. 

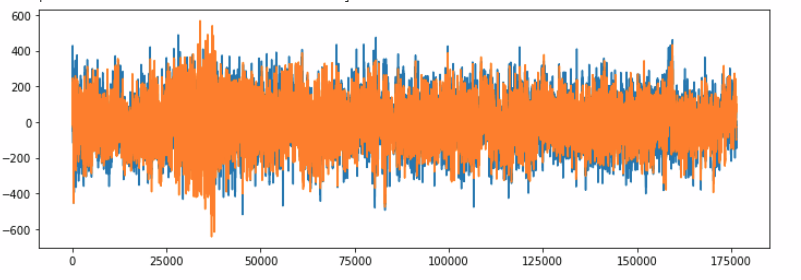
1. **Speech Separation :**   
   <https://github.com/philipperemy/timit>. This data set has audio samples contain human speech.   
   Within the **DataGenerator** folder are two Python scripts that create the dataset. It is assumed that a top-level folder exists called TIMIT\_WAV that contains the TIMIT dataset. The top-level folder should look something like this:  
   

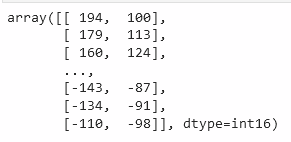
**3.2 EDA:**

Here’s a sample audio file waveplot. The Sample Rate of this waveplot is 44KHz. And its dual channel.

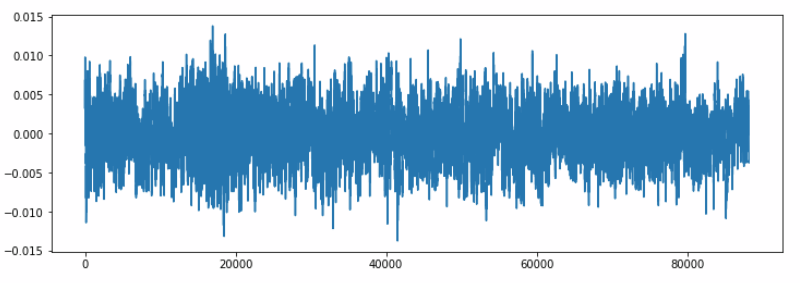
Sample Rate is the amount of data that can be heard/ stored in a second.

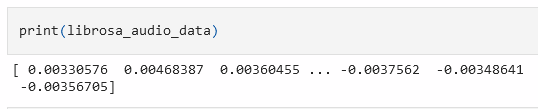
Dual Channel Audio has 2 channels that separate into different sound outputs.   
The particular file was loaded using scipy library.

****

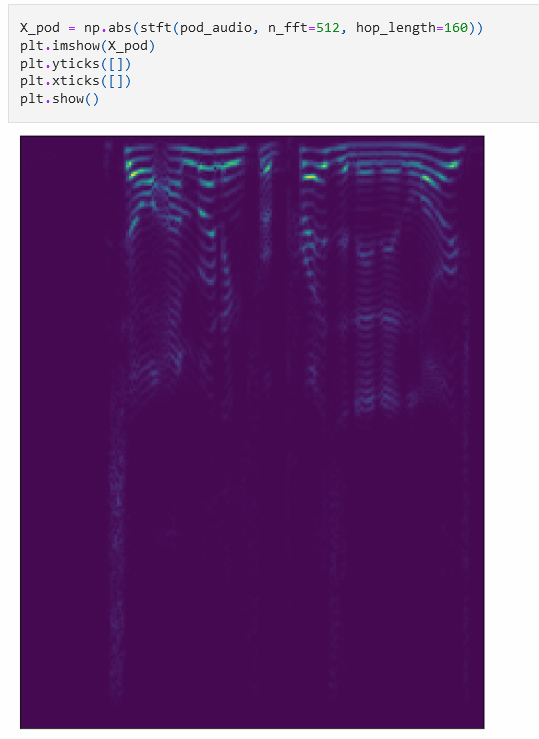
****

When a file is loaded using librosa library, the audio is sampled at 22KHz and it is converted into mono channel sound.

****

****

**Converting the audio data from time domain to frequency domain using Fourier transform. (here Short Time Fourier Transform)**

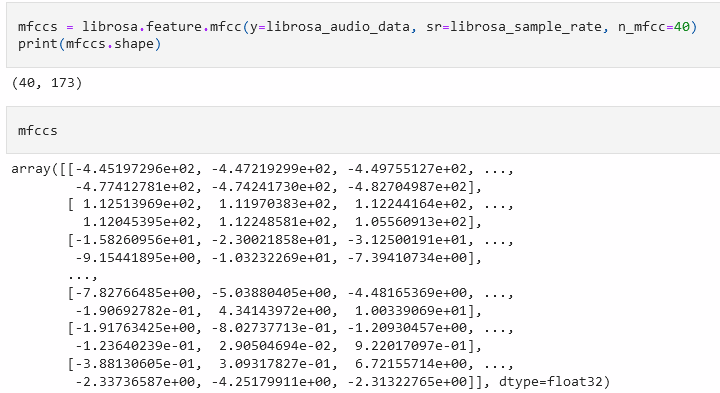
****

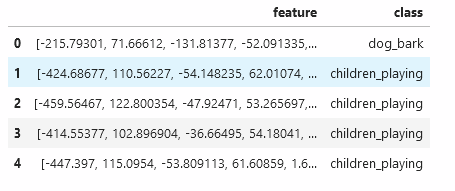
**3.2 Preprocessing :**

1. For Interference Separation :
   1. The podcast files needed to be in wav format for ease of use with pytorchaudio libraries as well as wav uncompressed format makes it much convenient to use the audio files.
      1. FFMPEG was used to convert the podcast files from MP3 to WAV which substantially increased the files sizes.
      2. The wav podcasts are then split in 8:2 train test splits.
      3. The wav files are divided in 5 sec audio files
   2. The interference files from ESC 50 Data are to be similarly placed in to respective train, validation directories for the open unmix library to mix the podcast files and interference noise.
2. For Speech Separation :
   1. The **datagenerator.py** script contains a class to create the data set. The dataset is saved as several pickle files. Each pickle file contains The pickle files are saved to a top level folder called **Data**.
   2. The **datagenerator2.py** takes the data from a given number of pickle files and feeds data into tensorflow session in batches.

**Feature Extraction :**

Here we will be using Mel-Frequency Cepstral Coefficients(MFCC) from the audio samples. The MFCC summarizes the frequency distribution across the window size, so it is possible to analyze both the frequency and time characteristics of the sound. These audio representations will allow us to identify features for classification.



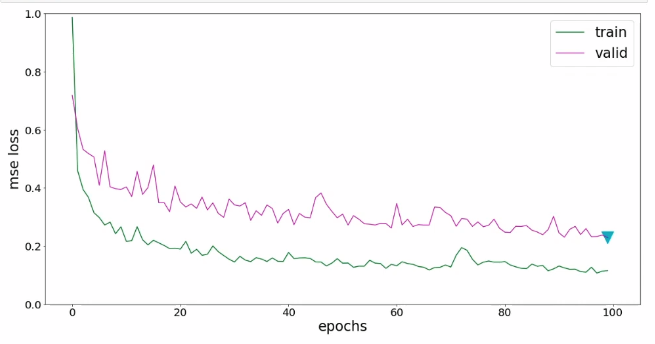


**3.3 Algorithms Used :**

1. For Interference Separation:  
    Here, we have used Open-Unmix for PyTorch for data generations as well as training the model.   
    Each *Open-Unmix* source model is based on a **three-layer bidirectional deep LSTM**. The model learns to predict the magnitude spectrogram of a target source, like *vocals*, from the magnitude spectrogram of a mixture input. Internally, the prediction is obtained by applying a mask on the input. The model is optimized in the magnitude domain using mean squared error.
2. For Speech Separation :  
    A simple **Feed Forward Neural network** has been implemented here using tensorFlow.   
    The network contains 2 hidden layers of 300 neurons and an output layer of 129 neurons (one for each frequency bin in the spectrogram). The output layer uses a sigmoid activation function. A mean squared error loss function is used on a known IBM. The following schematic represents the flow of code:

**3.4 Results :**

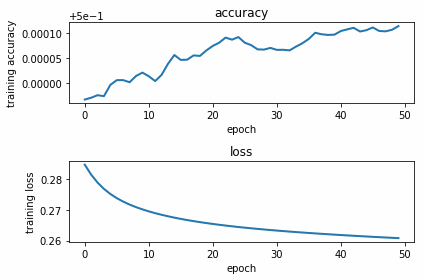
**For Interference Separation:**

****

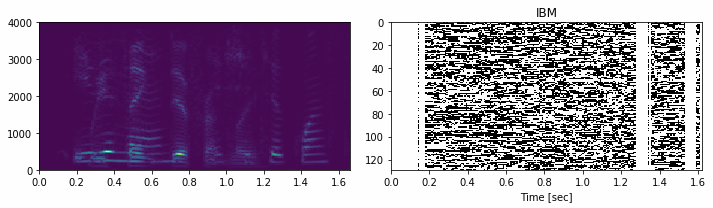
**The three-layer bidirectional deep LSTM implemented in the open unmix library seems to perform well for this use case.**

**For Speech Separation using Feed Forward Neural Network :**

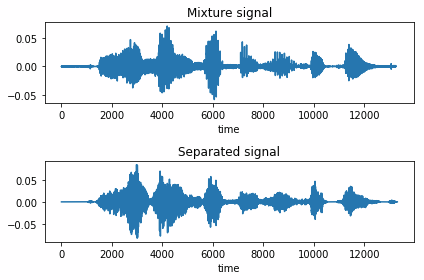
**After 50 epochs, the network struggles to find any pattern in the data. The accuracy after 50 epochs is still close to 50%.**

****

**A test signal containing mixture of 2 voices was fed into the network and the following IBM was produced:**

****

**After applying the IBM, the original sound wave looks (and sounds) the same as the original sound wave, implying that a feed forward network is not a good model for speech separation.**

****

**4. Conclusion and Results**

**4.1 Conclusion :**

Audio source separation can be hard to achieve however Deep Neural Networks are complex enough to make it possible. A slightly easier use-case such as Inference Separation can be achieved using a relatively simpler algorithm as we have demonstrated in this project.

However, Speech Separation requires an exponential complex Deep learning algorithm. A simple Feed Forward Network is not much competent for such use cases as is demonstrated here.

**4.2 Future Scope :**

Audio Speech Separation will require a much more complex Deep Learning algorithm such as Deep Clustering Combined with some other algorithm.

**3.3 References :**

[**Statquest With Josh Starmer**](https://www.youtube.com/channel/UCtYLUTtgS3k1Fg4y5tAhLbw)

[**Seth Adams**](https://www.youtube.com/user/seth8141)

<https://arxiv.org/ftp/arxiv/papers/1503/1503.06962.pdf>

[https://neuralnetworksanddeeplearning.com](https://neuralnetworksanddeeplearning.com/)

<https://github.com/tky823/DNN-based_source_separation>

<https://www.mathworks.com/help/audio/ug/cocktail-party-source-separation-using-deep-learning-networks.html>

<https://www.mathworks.com/help/audio/index.html>