Flipkart GRID 2.0

Solving for Voice Interactions in Indian Houses & Neighborhoods

Team Name : Stats Mafia

Institute Name: IISc Bangalore

Glossary

ASR	Automatic Speech Recognition
CNN	Convolutional neural network
CRNN	Convolutional Recurrent neural network
FCNN	Fully connected neural network
iSTFT	Inverse Short time Fourier transform
LSTM	Long Short term memory
ReLU	Rectified linear unit
RNN	Recurrent neural network
STFT	Short time Fourier transform
STLN	Signal transformation LSTM network

Use-cases

The proposed model can be added as an extra feature in the existing Flipkart App or on the website. Here are the list of the use cases in order of impact:

- An ASR system integrated with our proposed model can help the users navigate through the website while looking for products. It can help visually impaired users use the voice enabled navigation system in order to search the products
- The model can help the users in purchasing a product by placing the order through the voice enabled system
- Other features like cancelling an order, modifying an order et-cetera can also be driven by the proposed model
- The other features that are present in a General IT Helpline Desk can also be incorporated here

The main idea is that our proposed model will improve the efficiency of the ASR system by separating out the main speaker and de-noising the ambient noise present in the audio signal. Hence, all the use cases listed above will have better efficiency and will help in providing a seamless experience to the user.

Solution statement/ Proposed approach

We have divided the problem in 3 groups:

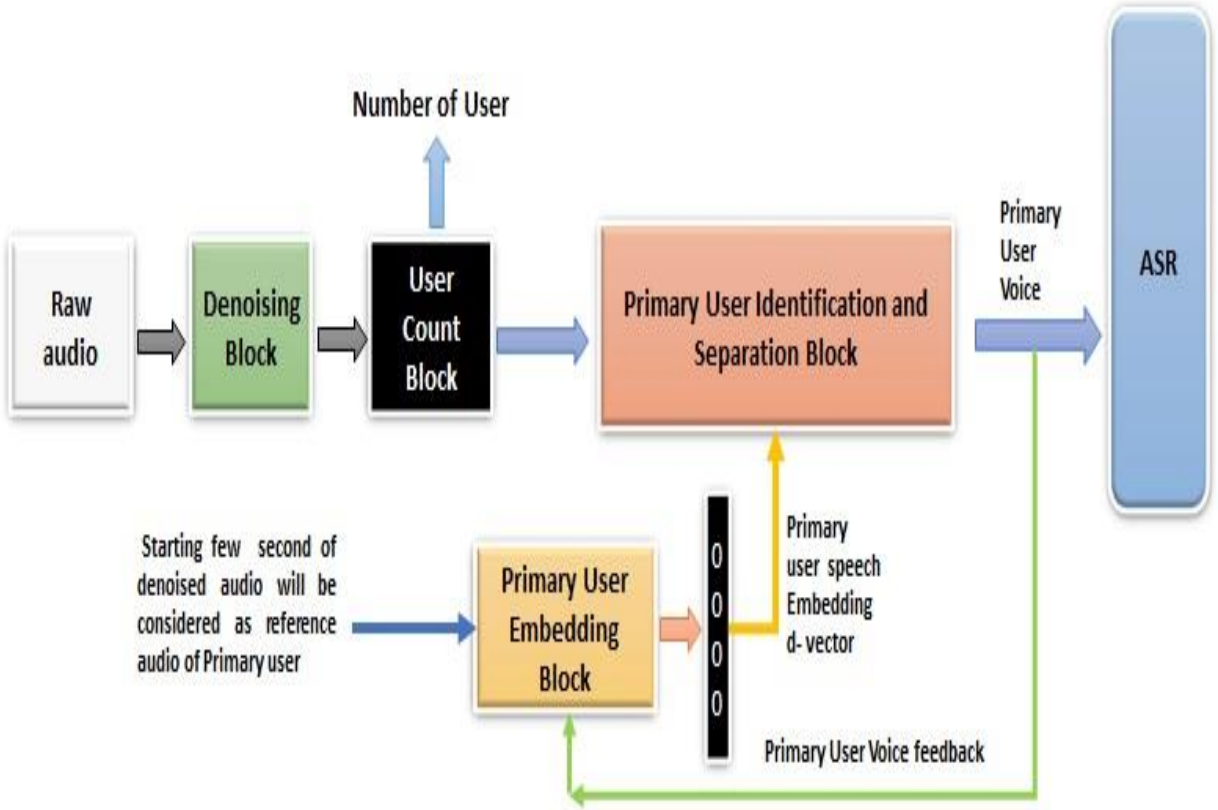
1. **Removing the background noise**: We will try to achieve this using STLN-based noise suppression De-noising block.

2. **Counting number of speakers in the audio**: CRNN based deep network will be used to detect the number of speakers. If only one speaker is detected, the signal will go directly as input to our ASR model.

1. **Primary User Audio Extraction:** This will be achieved by further dividing it into 2 sub-groups:

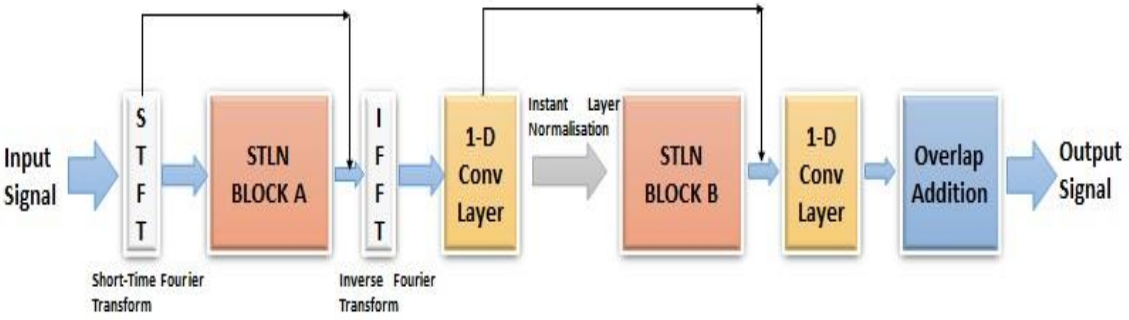
i. **Primary User Embedding block:** Here, we assume that it is the primary speaker who begins to speak in the audio and we take the first few seconds of audio of his interaction as input to this block and output will be the main user's embedding vector which will consist of the acoustic features of his voice.

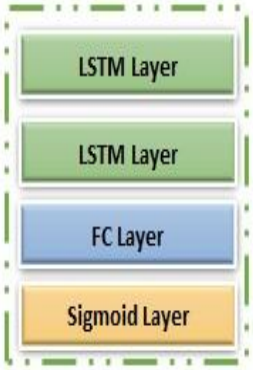
ii. **Primary User identification & Separation block:** We will be using a supervised approach to train our model. It will take the mixed audio as well as the primary user's embedding vector as the inputs and try to separate out the primary user's audio.



De-Noising Block Architecture

- This block will take the noisy mixed audio signal as input and try to remove the ambient noise (if any) present in the audio without suppressing the users' voice.
- Log power spectrum from noisy signal is extracted using STFT and is passed through STLN Block which provides real time capability detection of one frame at a time.
- The predicted speech signal is reconstructed using the estimated magnitude from STLN block and phase of noisy mixture.
- The FC layer along with Sigmoid layer within the STLN block is used to create masking that suppress noise, while 1-D Conv layer is used to create feature representation.
- In last step, the signal is reconstructed using overlap and add procedure.
- The proposed model configuration will have roughly 128 units in each LSTM layers for 2 STLN Blocks.
- The frame size or window size of 32 ms will be used for audio signal that will be shifted at a rate of 8 ms.
- 256 filters of 1-D convolution layer will be used, and dropout with probability 0.25 will be tried initially.
- The output from this block(de-noised mixed audio signal) will be sent to the speaker count block to detect if there is a single speaker or multiple speaker.



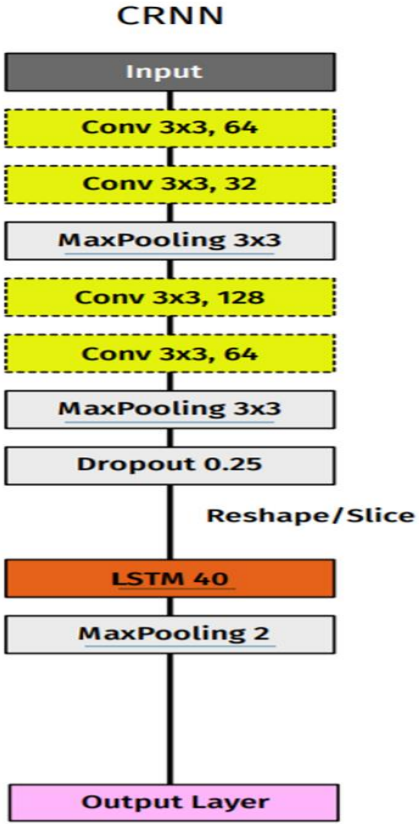


STLN consist of 2 layers of LSTM followed by 1 layer of Fully Connected Layer and output a sigmoid layer. These layers formed stacked network architecture.

STLN or Signal Transformation LSTM Network Block Diagram

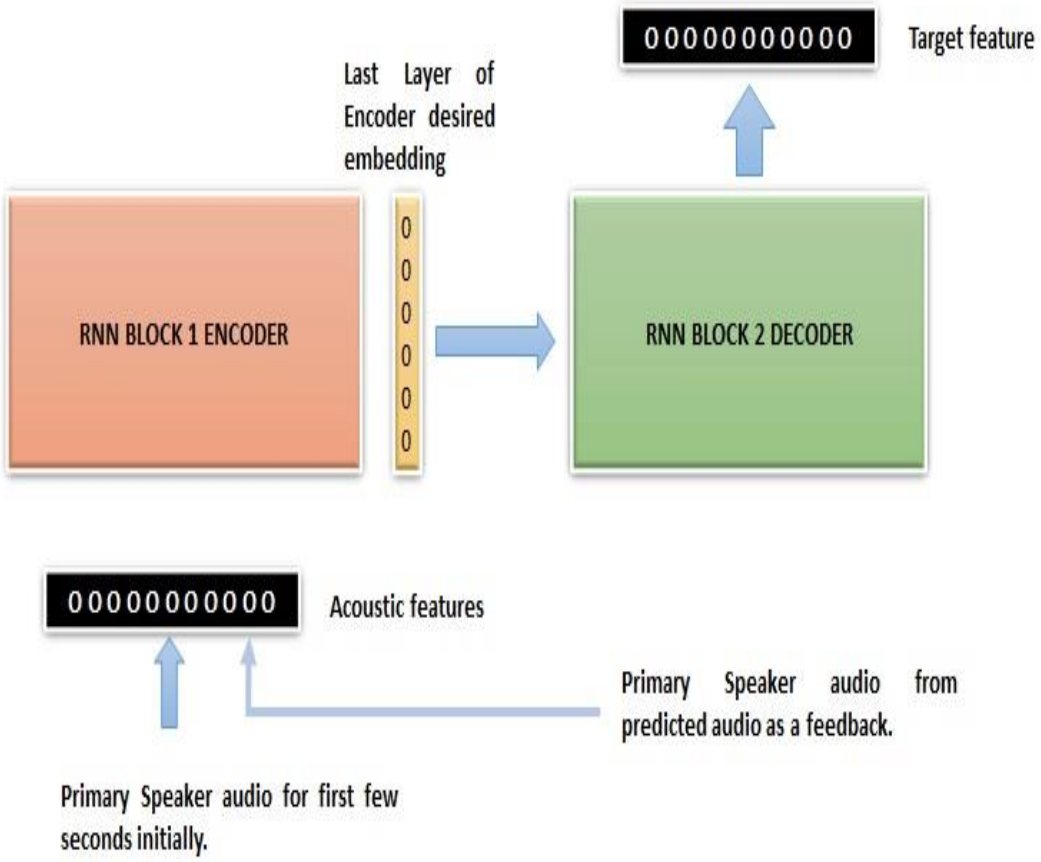
User Count Block Architecture

- This block of the system will take audio signal as the input and will keep counting the number of speakers present in the audio in every short time scale to aid real-time processing.
- We will be using state of the art Convolutional Recurrent Architecture to carry this out.
- We are using a supervised learning approach here to train the model.
- Convolutional Layers excel in capturing local structures whereas RNNs can detect structure in sequential data of arbitrary length, which makes RNNs to model time series.
- A CRNN is hence used to get the best features out of the two models.
- The output from this deep learning net will help us identify whether there are multiple speakers or if there is only a single speaker and the output will be fed into another deep learning net to identify the primary speaker if there are multiple speakers identified from the Audio Channel.

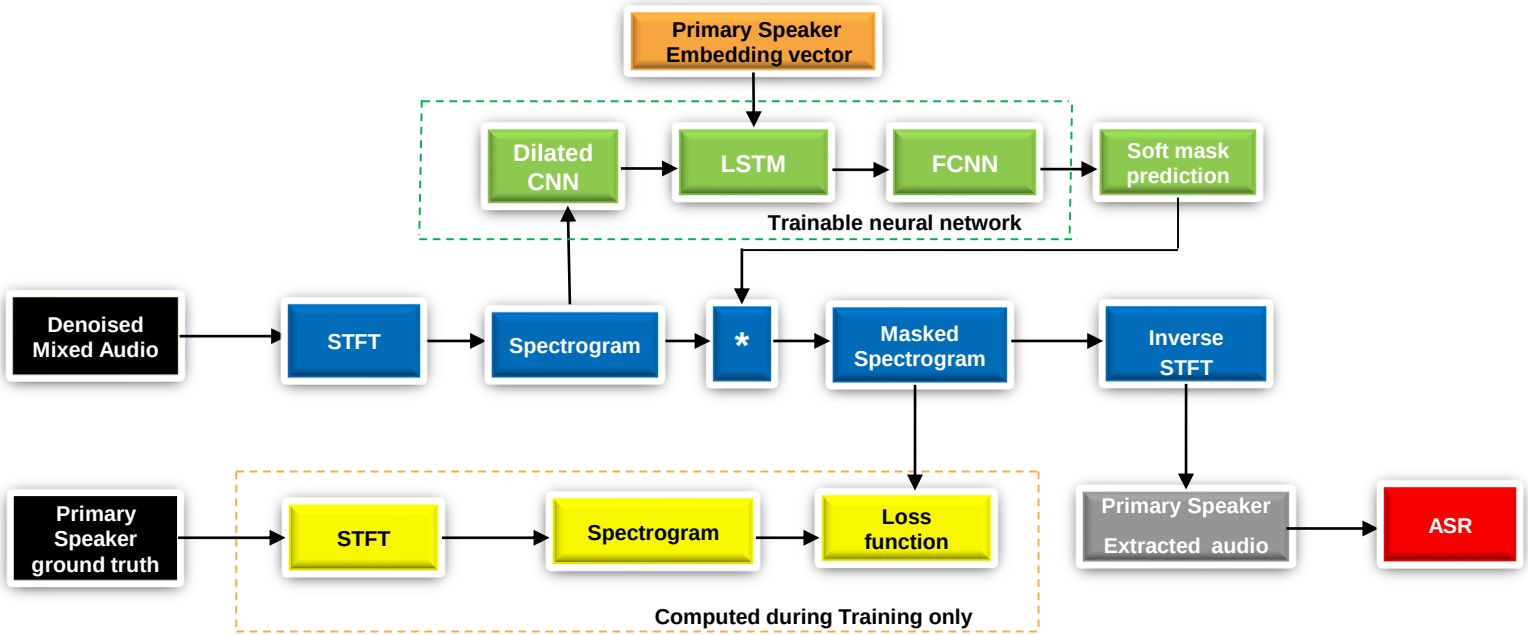


Primary User Embedding Block

- It will take only the primary user voice signal as the input which can be obtained by the first few seconds of the audio where primary user interacts with the model.
- In real life application, we can think of this as first few words uttered by the main user who wants to use the App.
- The Embedding from audio segment of primary speaker is generated using dual RNN (LSTM) auto encoder.
- The Auto Encoder consists of an RNN based Encoder which reduces the dimension of the acoustic features and in the process learns any latent or hidden information along.
- This reduced dimension vector obtained is the desired encoded d-vector of the primary speaker audio segment.
- The decoder which also is formed of RNN converts this vector back to original target feature.
- The goal is to predict target feature as close as acoustic feature.
- Loss is calculated using mean square difference between the acoustic feature and target feature.
- The idea is to first generate the embedding of audio from initial few seconds of audio interaction of primary speaker, there after the embedding are updated from the predicted audio of the primary speaker given back as a feed back.
- The output from the block will be sent to the primary user separation block to extract the primary user's audio data.



Primary User Identification and Separation Block



Aims to identify and separate the voice of the primary speaker from multi-speaker signals, by making use of the first few seconds of the audio where the primary speaker invokes the voice-based conversation assistant can be used. Every time, a user wants to use the App, his/ her first few utterances will be considered as the voice of the main user. This method would work for new as well as returning user.
Two separate neural networks is required to achieve this:

- A speaker recognition network that produces speaker-discriminative embeddings
- A spectrogram masking network that takes both noisy spectrogram and speaker embedding as input, and produces a mask

Core Components of the Primary Speaker Separation Block

- **Short-time Fourier transform(STFT) on the mixed audio :** Fourier-related transform used to determine the sinusoidal frequency and phase content of local sections of a signal as it changes over time. It provides the time-localized frequency information for situations in which frequency components of a signal vary over time. A 25 ms Hann window and a 10 ms hop size will be used to compute the STFT. Using this, we can derive input features, e.g. the compressed magnitude spectrogram.

- **Short-time Fourier transform(STFT) on the Primary speaker clean audio :** It is used to derive input features of the primary speaker clean audio (ground truth) to create the compressed magnitude spectrogram.

- **Trainable Spectrogram masking Neural Network:** It will take two inputs: a vector of the primary speaker embedding, and a magnitude spectrogram computed from a noisy audio. The network will be trained to minimize the difference (L2 Loss with all spectra being power-law compressed) between the masked magnitude spectrogram and the primary speaker magnitude spectrogram computed from STFT applied on the ground truth of the primary speaker audio. The network will roughly consist of 8 dilated convolutional layers with 64 nodes in first 7 layers and 8 nodes in the last layer, 1 LSTM layer with 400 nodes, and 2 fully connected layers with 600 nodes in each layer, to capture low-level acoustic features more effectively, each with ReLU activations except the last layer, which will have a sigmoid activation. The network will predict a soft mask.

- **Soft mask prediction :** The key idea behind soft mask-based prediction is to estimate a spectrographic soft mask to suppress the unwanted spectrogram components. In soft mask-based SE methods, each time-frequency component is assigned a probability linked to the primary speaker's speech. This output will be element-wise multiplied with the magnitude spectrogram of the mixed signal to produce an enhanced magnitude spectrogram.

- **Inverse Short-time Fourier transform(iSTFT) on soft mask enhanced spectrogram:** The inverse STFT of the masked STFT will be computed to generate an enhanced time-domain output. To obtain the enhanced waveform, the phase of the mixed audio will be directly merged to the enhanced magnitude spectrogram and inverse STFT is applied on the result. This will give us the primary speaker audio data without any speech utterances by other speaker. This extracted output signal can be used as the input to our Automatic Speech Recognition System(ASR).

Approximate Computational Complexity

De-Noising Block: This block consists of two STNL blocks where each STNL block contains 2 stacked LSTM layers and one Fully connected network.
Computational complexity per unit time of a LSTM layer is O(W) where W is the number of weights.
C1:Computational complexity per unit time of 2 STLN= O(2(128^2*4*2+128*64*4*2+128*64+128*3*2+64*64))

User Count Block: Consists of 4 Conv layers and 1 LSTM layer.
C2:Computational complexity per unit time= O(64*3^2*64+64*3^2*32+32*3^2*128+128*3^2*64+64*40*4+40^2*4+40*64+40*3)

Primary User Embedding Block: Consists of 2 LSTM network.
C3:Computational complexity per unit time= O(2(100^2*4+100*39*4+100*39+100*3))

Primary User Identification and Separation block: Consists of 8 CNN, 1 LSTM, 2 FCC.
Computational complexity per unit time of CNN layers= O(64*1*5^2+6(64*64*5^2)+64*8*5^2)
Computational complexity per unit time of LSTM layer= O(400^2*4+400*64*4+400*64+400*3)
Computational complexity per unit time of FCC layers= O(64*600+600^2)
C4:Computational complexity per unit time for this block = O(64*1*5^2+6(64*64*5^2)+64*8*5^2+ 400^2*4+400*64*4+400*64+400*3+ 64*600+600^2)

Considering the worst time complexity, we can ignore the primary user embedding block complexity as it is run parallelly and its complexity is lesser than user count block. So, we have De-noising block, user count block and primary user identification & separation block which are stacked sequentially.
Therefore, approximate total computational complexity per unit time for our model will be sum of C1,C2 and C4.

Limitations

- We are considering the user who is initiating the conversation with the model is the primary user, we are using the first few seconds of the audio to generate features related to primary speaker. We are assuming that no secondary user is interrupting the audio in first few seconds.

- All architecture need large training data in order to achieve good accuracy.

- Extremely high noise of range greater than 40KHz will be suppressed by the model and will not be counted by the user count block.

Future Scope

- We would like to create primary user embedding using the energy observed by the system, that is for first initial seconds we will consider user having highest energy as primary user.
- We would like to integrate the model in such a way that entire model is trained jointly rather than in a cascading manner.
- Convert the model to be self-supervised / unsupervised.


Flipkart
GRID 2.0