Multi-channel U-Net for Audio Source Separation

Nakula Neeraje (190525)
Electrical Engineering
IIT Kanpur
Kanpur, India
nakula@iitk.ac.in

Videh Aggarwal (190960)
Electrical Engineering
IIT Kanpur
Kanpur, India
videhag@iitk.ac.in

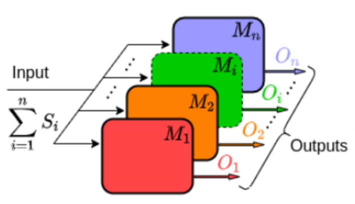
***Abstract*—Audio source separation problems were conventionally dealt by using separate models to identify each source. Later on, conditioned U-nets (C U-nets) came up which trained a single model using control units and parameters to separate audio sources. In this report, we have used a multi-channel U-net (M U-net) for the audio source separation task. We have utilized energy based weighting (EBW) strategies in our loss function to prevent the model from preferring sources with higher energy levels. Our model has a lower training cost compared to the previous two mentioned, and also utilizes parameter sharing. This comes as an added advantage since our results for bass, drums and the remaining sources are at par with the above two models mentioned.**

I. Introduction

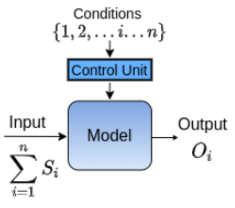
Audio source separation, and in specific music source separation deals with estimation and/or reproducing the individual sources when given an audio mixture. This has many applications and is relevant to the music industry as a whole. Unmixing music and music education can be widely benefited with development in this field.

We train a model to output the different sources present in the input when given the input mixture in a certain representation. The input could be in the time-domain representation [1] or in the form of Spectograms that are in the frequency domain [2]. The Spectograms are a downsampled version of the original time domain waveform. Hence, Spectograms would need small convolution kernels because they have lower sampling rates. Similarly, the number of trainable parameters would be lower. Models employing Spectograms generally output ”masks”, that can be element-wise mulitplied with the mixture Spectograms to output the individual source Spectograms. Convolutional Neural Networks(CNNs) and Long short term memory(LSTM) are some popular models employed for Audio Source Separation. For CNNs in specific, U-Net based architectures are very popular because they are simple to train.

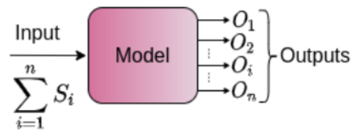
Broadly, there have been 3 different attempts at solving music separation problems. The type (a) method involves making independent models for each source. Although easy to understand, each model has its own set of parameters amounting to a huge total. Also, no parameters are being shared which leads to lack of mutual information between the sources. To overcome these problems Meseguer-Brocal and Peeters came up with the type (b) system : C U-Net. It uses only a single U-net to predict a source in addition to a neural network called the ’control unit’. A one-hot encoded input, representing the source we wish to isolate, is given to the control unit. This outputs certain control parameters which influence the U-net model to generate the desired audio source. But, since we would have to feed a one-hot encoded input for every audio source present (say K), we would have to train this model K times for each data sample per epoch increasing the total number of iterations.



(a) Dedicated Models



(b) Conditioned Model



(c) Multi-task Model

Fig. 1: 3 different approaches

The type (c) system M U-net discussed in this report solves all the above problems. It has the standard U-net architecture but with multiple ’channels’ in the output used to generate the multiple sources. Since it is a single model, the problem of huge number of parameters and lack of mutual information is resolved. Also, each data sample is trained only once per epoch like a standard U-net thus reducing the total iterations and the training cost. We use a multi-weighted loss function to separate the audio mixture into 4 sources - vocals, drums, bass and the rest; this could very well be extended to any number of sources.

II. Proposed Method

Our M U-Net model takes as input a log magnitude spectrogram of the audio mixture and outputs four masks corresponding to four sources - Vocals, Drums, Bass and the

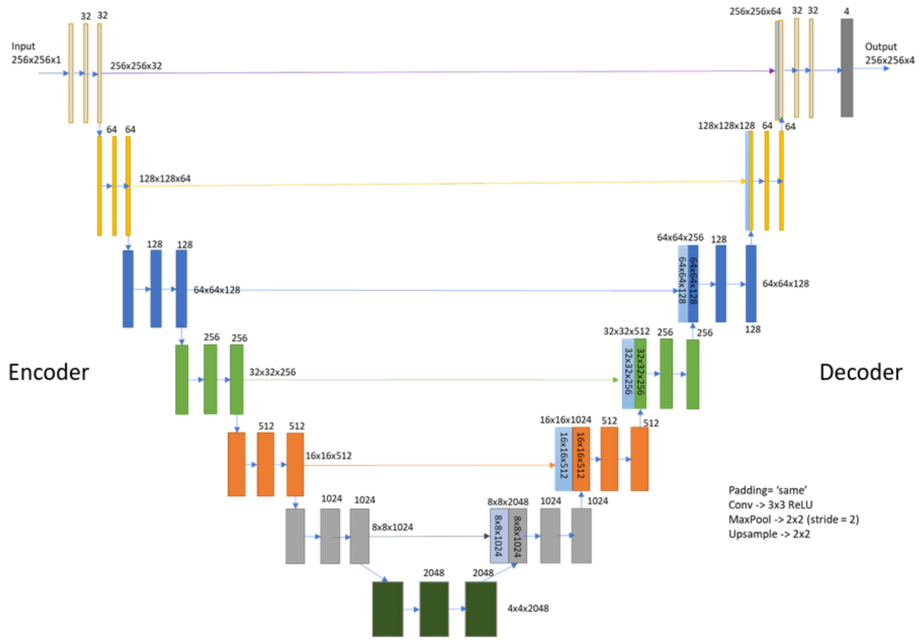


Fig. 2: Model Architecture

rest. We have defined a loss function L_i for each source as below:

$$L_i = \sum_{n=1}^{T} \sum_{m=1}^{F} |S_i(n,m) - M_i(n,m) S_{mix}(n,m)|. \quad (1)$$

Here n and m are variables for the time and frequency bins respectively. $S_i(n,m)$ represents the ground truth spectrogram and $M_i(n,m)$ the predicted mask for the i^{th} source. $S_{mix}(n,m)$ represents the audio mixture spectrogram. In effect, L_i is the L1 norm.

We would also need a consolidated loss that combines the individual loss in a meaningful manner. We choose to use the weighted sum of the individual sources as the consolidated loss:

$$\mathcal{L} = \sum_{i=1}^{K} w_i L_i, \quad (2)$$

We take K to be the number of individual sources present in the input mixture across the data samples.

A. Loss weighting scheme

We use an Energy based strategy to weight the losses. Heuristically, sources with greater median energy would be better estimated in our model. To repress the bias, we assign weights to neutralize the advantage. The Energy corresponding to each source has been calculated by squaring and summing up the individual bins in the Ground-Truth Spectograms corresponding to the different sources. We then take the median Energy across all data samples.

$$w_i = \max_{j \in \{1,.......,K\}} E_j / E_i \quad (3)$$

In this scheme, we observe that the source with the highest median energy would have $w_i = 1$, with every other source having a weight > 1. Furthermore, the source with the lowest median energy would have the greatest weight in the Loss function.

III. Dataset

We use the Musdb18 dataset for our task. It consists of 150 different audio mixtures, out of which 100 are taken to be training and 50 as the testing samples. We downsample the audio mixtures to 10880 Hz and split them into pieces of 6s length. Then, we convert these pieces to 512x256 spectrograms by applying STFT (Short-time Fourier Transform) on them using a 'Hanning' window of size 1022 and hop-size of 256. Finally, we warp these spectrograms to 256x256.

IV. Network Architecture

We use a standard U-net architecture with filter sizes 32, 64, 128, 256, 512, 1024, 2048. It consists of an encoder and a decoder path, and has 6 downsampling (maxpooling) layers and 6 upsampling (transposed convolution) layers. The encoder downsamples the original spectrogram, producing deep features and keeping the information that is relevant for source separation. Since the dimensions are now reduced, the decoder restores the original dimensions in the output by upsampling. 'Skip connections' between the encoder and the decoder improve this upsampling by progressively providing finer-grained information to the decoder. They also mitigate the issue of vanishing gradients, which can creep up in really deep networks.

The model is trained using the 'Adam' optimizer and a dropout of 0.1. The input to it is the log magnitude spectrogram of size 256x256. The output consists of 4 channels, each designed to produce a mask of size 256x256. The total number of trainable parameters are 467,076 proving that the training cost is cheap. The model architecture is shown in fig 2.

V. Evaluation and Results

We choose to evaluate the Signal to Distortion ratio(SDR) for evaluating the performance of our model on Music Source Separation performance.

$$\hat{y} = y_{target} + \epsilon \tag{4}$$

$$S.D.R. = 20 \log \frac{||y_{target}||_2}{||y_{target} - \hat{y}||_2} \tag{5}$$

We evaluate this metric for each individual source and across our samples in the test set.

TABLE I: S.D.R evaluation for examples in the test set

Examples	Vocals	Drums	Bass	Rest
Test1	-57.88	15.86	9.17	5.81
Test2	0.19	9.73	19.16	2.47
Test3	0.10	-0.01	-0.01	5.51
Test4	0.68	20.22	13.07	5.34
Test5	0.02	17.64	8.90	5.60

VI. Conclusions

As can be seen from the S.D.R evaluation for the test samples, our model has got good results for drums, bass and the 'rest' sources. This point is further emphasized by the spectrogram plots of test samples in the following pages. Unfortunately our vocals S.D.R values are low. A possible solution to this is to find vocals by subtracting the sum of masks of all the other sources from a matrix completely filled with ones. This would give nice results for vocals too.

Another look at the table above tells us that our results for Test3 are bad. This was because it had silent sources, and our model could not figure out how to generate these. This is something which we are interested to work on in the future.

References

[1] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," 2018. [Online]. Available: https://arxiv.org/abs/1806.03185

[2] N. Takahashi and Y. Mitsufuji, "Multi-scale multi-band densenets for audio source separation," 2017. [Online]. Available: https://arxiv.org/abs/1706.09588

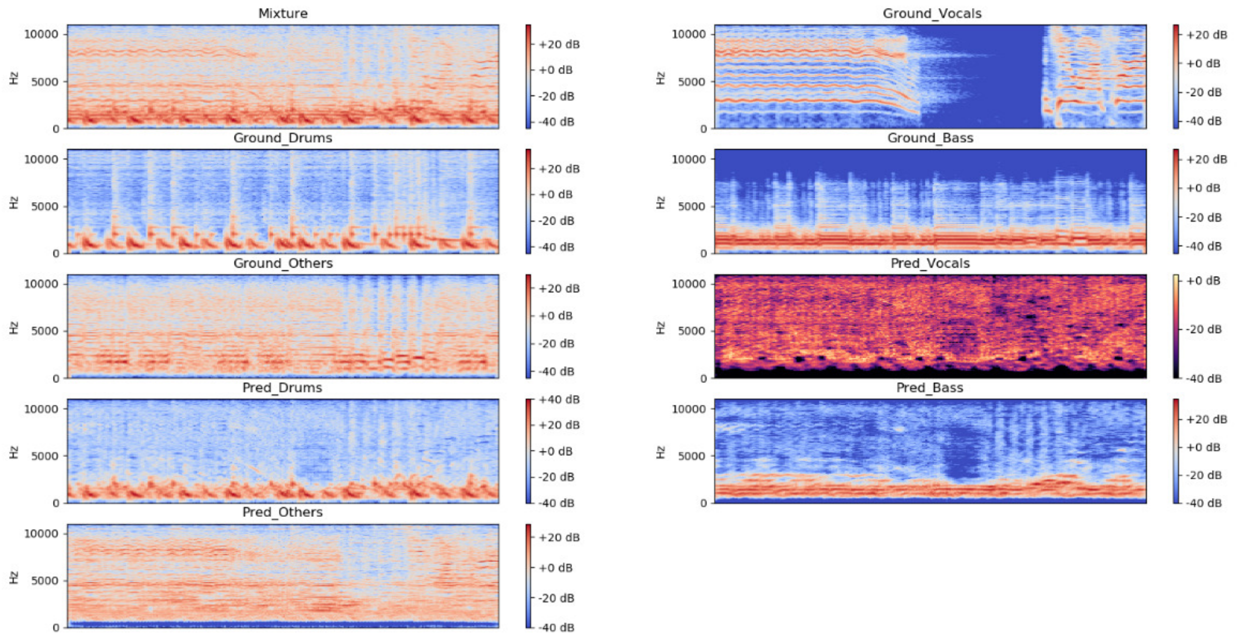


Fig. 3: This is the spectogram for test1

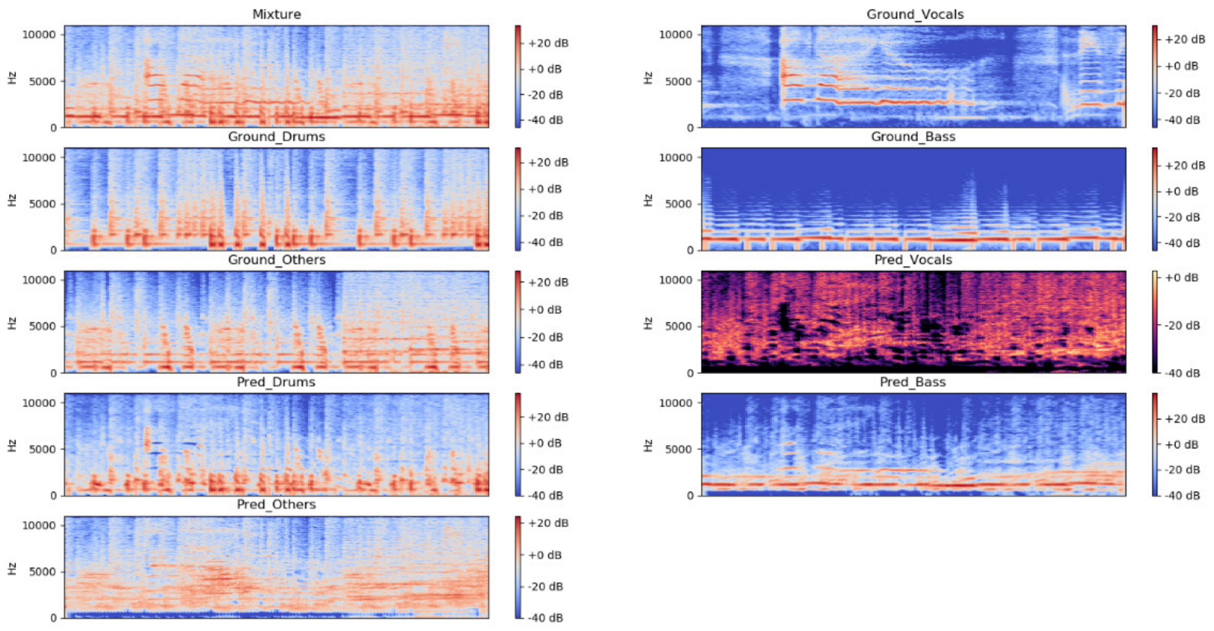


Fig. 4: This is the spectogram for test2