Monaural Source Separation Using Neural Networks

Simon Kim
Stanford University
Computer Science
spkim@stanford.edu

Mark Kwon
Stanford University
Computer Science
hjkwon93@stanford.edu

Sunmi Lee
Stanford University
Computer Science
sunmilee@stanford.edu

Abstract

Source separation of audio signals is very applicable in many real-world situations, such as enhancement of accuracy in speech recognition. Monaural source separation in particular is a challenging problem because there is only a single channel of information available and there are many possible solutions. Here, we focus on monuaral source separation using neural networks to enhance separation performance. We propose a joint optimization of masking layers and deep learning models in order to enhance the model of previous works. We use the DAPs dataset for clean recordings of human speech and evaluate the model's performance using the BASS performance measures such as SIR, SAR and SDR. Our model achieves about 15dB SIR, 7-8dB SAR/SDR gain for the clean signal compared to our baseline model.

1 Introduction

Reducing or eliminating background noise has many real-world applications because noise in the dataset introduces difficulties to spoken language tasks such as transcription of spoken language to phonemes or machine comprehension of human speech. To tackle this challenge, there have been many previous approaches including non-negative matrix factorization and a joint optimization of recurrent neural network and deep neural network. However, current separation models are still far behind human capability, with monaural source separation being especially difficult because only single channel signal is available.

The goal of this paper is to introduce an efficient method to separate, denoise, and emphasize specific sound sources using neural networks and masking. In this paper, we explore the use of a DNN and an RNN for monaural speech separation, with the joint optimization of the network with a hard masking function. We use the DAPS (Device and Produced Speech) dataset, which has recordings of male and female speakers in a studio setting, and combine noise collected to produce the mixed sources. We use deep learning models such as DNN and RNN with hard masking to do the initial separation of the mixture to clean and noisy sources. Then, we introduce further enhancements such as frequency-based loss, Bi-Directional GRU and masking iterations to further enhance the clean speech.

2 Related Work

There have been many prior approaches to monaural source separation. Non-negative matrix factorization (NMF) is one of the most well-known techniques that are used to separate distinct sources from a mixture. In this approach, a non-negative data matrix is approximated by a product of a basis matrix and an encoding matrix with non-negative elements. [4] The fundamental assumption underlying the conventional NMF technique is that the subspaces which the separate sources span are almost orthogonal to each other. However, target source separation with the concatenated basis matrix turns out to be problematic if there exists some overlap between the subspaces that the bases for the individual sources span. Probabilistic latent semantic indexing (PLSI) is also widely used. Similar to NMF, it factorizes time-frequency spectral representations by learning the nonnegative reconstruction bases and weights. [2]

NMF and PLSI models are linear models with nonnegative constraints. [2] Each can be viewed as one linear neural network with non-negative weights and coefficients. In order to further enhance the method, there are approaches that use non-linear models, such as Deep Neural Networks (DNN) and Recurrent Neural Networks (RNN). One approach is modeling the training data for the source signals using a single deep neural network (DNN). The DNN is used as a spectral domain classifier which can classify its input into each possible source type. [1] Another approach is to use both DNNs and RNNs rather than using a single deep learning model. In addition, some prior works also add in a joint optimization of the deep learning models and masking which enforces a reconstruction constraint. [3] These models that use deep learning models with masking achieve a higher success metric when compared to the NMF baseline. In these deep learning models, instead of using a spectral representation for separation directly, the output layer reconstructs the spectral domain signals based on the learned hidden representations. [2]

While it is not exactly the same, there are also denoising-based approaches, which are similar in the sense that they utilize deep learning based models to learn the mapping from the mixture signals to one of the sources among the mixture signals. Maas et al. [5] proposed to apply a DRNN to predict clean speech features, which is fundamentally the same with noise separation. The difference is that denoising methods do not consider the relationships between target and other sources in the mixture, which is necessary for noise separation.

3 Dataset

We acquired dataset of clean speech of 20 different speakers, 10 female and 10 male, from DAPS (Device and Produced Speech) dataset, which included recordings of each speaker in a studio-setting free of any background noise. With these, we added background noise by combining their sound waves with those of different noises, such as raindrops, jungle sounds, cafe, and factory sounds.

When using them for training our neural network model, the mixed sounds were the inputs, and the targets were the two sounds we used to make the inputs.

3.1 Fourier Transform of Raw Sounds

To preprocess the data before feeding it into the neural network model, we performed Short Time Fourier Transforms to create spectrogram representations. We applied the hamming window and set the timeframe to be 10ms. Using spectrogram of sound files allowed the model the flexibility to incorporate frequency-based loss function, as will be discussed in next sections. We applied inverse Fourier Transform to return the spectrograms back to sound files after the model completed running.

The spectrograms of clean, noise, and mixed sounds are as follows:

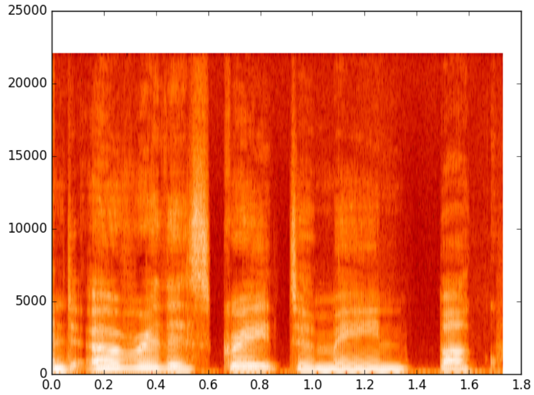

Figure 1: Spectrogram of Clean Sound

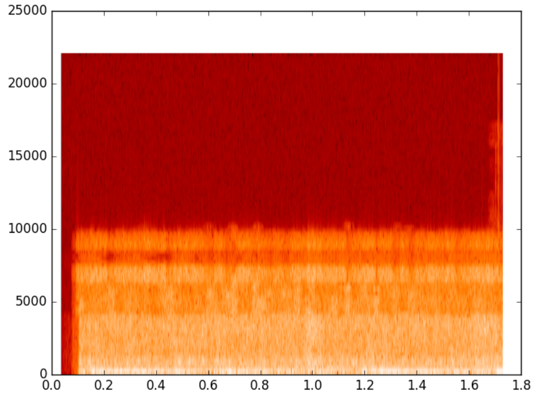

Figure 2: Spectrogram of Noise Sound

3.2 Minibatch and Paddings

The model was fed in data in minibatches of size 16. As a prerequisite to running the neural network model, all inputs must have the same length and dimensions. Therefore, within each batch we normalized the lengths of inputs to match that of the longest sound file. In average, each padded sound file had length of around 2 seconds.

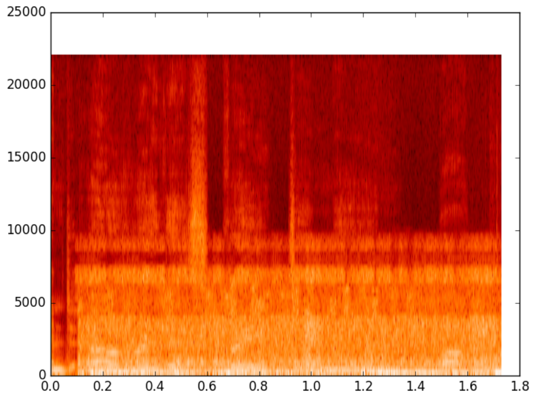

Figure 3: Spectrogram of Mixed Sounds

4 Model

4.1 Architecture of Neural Network Model

In training of the model, the preprocessed data was fed into a neural network model, and the target was a horizontal concatenation of spectrogram matrices of clean and noise sounds. Whereas the input data would be a spectrogram of the mixed sounds, the output data would be two spectrograms of the separated sounds concatenated right after one another to form a single matrix of twice the length.

From the output data of the model $\hat{y_1}_t$ and $\hat{y_2}_t$, we created masks of clean and noise sounds. A binary time-frequency mask, which we identified as a hard-mask, was calculated to be as follows:

$$M_h(x) = \begin{cases} 1, & \text{if } |\hat{y_1}_t(x)| > |\hat{y_2}_t(x)| \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

The idea behind hard-masking is that, for a binary classification task of discerning clean and noise sounds from a mixed one, we could assign each frequency bin of a spectrogram of the mixed to a class that has greater value for that bin. In a way, hard-masking splits the spectrogram of a mixed sound into two based on the output of the model, where an element with corresponding mask value of 1 retains the original value but that with mask value 0 is reduced to zero after masking.

A soft-mask was calculated similarly, except rather than making the mask binary, we calculated the ratio of each of the corresponding elements as follows:

$$M_s(x) = \frac{|\hat{y_1}_t(x)|}{|\hat{y_1}_t(x)| + |\hat{y_2}_t(x)|} \tag{2}$$

Soft-masking differs from the hard-masking because if two elements are similar in value, one is not discarded entirely as was the case for the hard-masking. In evaluations we will discuss how hard and soft-masking performed on different models using the metrics discussed below.

After the masks were obtained, we applied them to the original file of mixed sounds to separate the clean and noise sounds.

$$\begin{aligned} \hat{s_{c_t}}(x) &= M(x) \odot S_t(x) \\ \hat{s_{n_t}}(x) &= (1 - M(x)) \odot S_t(x) \end{aligned} \tag{3}$$

where s_t represents spectrogram of the mixed sounds, s_{c_t} and s_{n_t} represent spectrograms of separated clean and noise sounds, and $\odot$ is the hadamard (element-wise) product of two matrices. We then applied Inverse Fourier Transforms on outputs of the masking to recreate sound files.

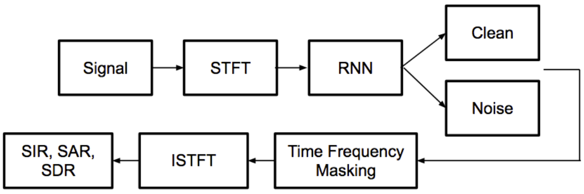

Figure 4: Overall Architecture of Model

4.2 Baseline Model

As a baseline model, we implemented a single-layered, one-dimensional neural network model with only l_2 norm loss function. From this baseline we tested with different parameters such as number of layers, using frequency-based loss, and models such as a bidirectional RNN and masking iterations.

4.3 Frequency-based Loss

Using a unnormalized l_2 norm as a loss function may measure how far the output matrix is from the target, but it does not accurately represent reality where humans can only hear a limited bandwidth of frequencies. As a result we devised a loss function that applied more weights to spectrogram bins of meaningful frequencies that are distinguishable by human ears.

We used absolute threshold of hearing (ATH), which defined that the relationship between energy threshold and frequency in Hertz is approximated to be as follows:

$$
\begin{aligned}
ATH(fq) = 3.64(\frac{fg}{1000})^{-0.8} \\
- 6.5e^{-0.6}(\frac{fg}{1000} - 3.3)^2 \\
+ 10^{-3}(\frac{fg}{1000})^4
\end{aligned} \tag{4}
$$

Figure 5 below shows the distribution of weights.

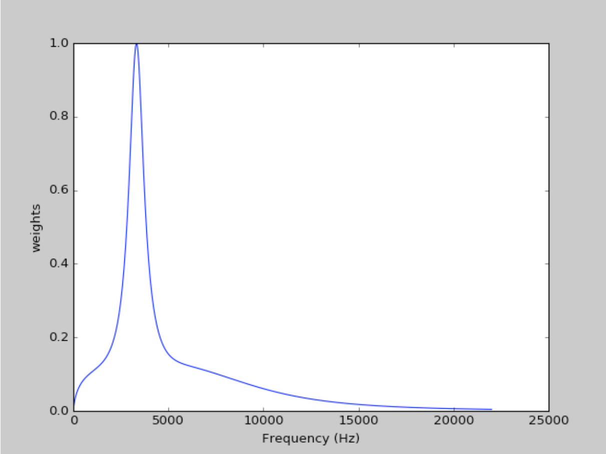

Figure 5: Frequency-based Loss Weights

4.4 Bidirectional GRU RNN

Compared to the baseline of a single-layered one-directional neural network model, we wanted to see how a bidirectional GRU RNN performed. The evaluations section discusses how different parameters we tested performed on different models.

4.5 Masking Iterations

After each iteration of a neural network model run, we obtain a mask that we can apply to the mixed sounds to create clean and noise sounds. Our intuition was that by applying it to the mixed sounds to amplify the frequencies of the target sound and feeding it again into the neural network model, we would obtain a new mask that can perform better separations. Therefore, in each iteration we applied the mask from the previous iteration, which was multiplied by a factor of 0.1, to the input sound file to amplify the clean sounds, and ran the model again until the mask converged. Our intuition was that this would allow the model to highlight specific parts of mixed sounds so that the final mask would perform noise separation much better. The results will be discussed in the following sections.

5 Analysis Metrics

We evaluated our model using three metrics: SDR (Source to Distortion Ratio), SIR (Source to Interference Ratio), and SAR (Source to Artifact Ratio). The metrics can be explained as follows:

Assume we have a target sound s_{target}. With the estimated target $\hat{s}$, we can decompose $\hat{s}$ into s_{target}, e_{interf}, e_{noise}, e_{artif}, the first parameter indicating the target sound we wish to accomplish and the latter three representing error terms corresponding to interference, noise, and artifacts, respectively. These error terms include noises from unwanted sources, sensor noises, and distortions and/or artifacts. By calculating the energy ratios of these components within the predicted outcome sound, we can numerically evaluate how well the noise separation task has performed. The model compares the clean/noise outputs from masking with the target sounds to obtain these metrics in the following manners:

SDR (Source to Distortion Ratio) calculates how much total distortion exists in the predicted sound. Higher value of SDR indicates there is less distortion noises overall, which include the interference, noise, and artifact errors.

$$
SDR := 10 \log_{10} \frac{\left\|s_{target}\right\|^2}{\left\|e_{interf} + e_{noise} + e_{artif}\right\|^2} \tag{5}
$$

SIR (Source to Interference Ratio) directly compares how well the noises from unwanted sound sources have been separated from the target sounds. Higher value of SIR indicates the target sound is well-distinguished from the interference noises that we wish to separate.

$$
SIR := 10 \log_{10} \frac{\left\|s_{target}\right\|^2}{\left\|e_{interf}\right\|^2} \tag{6}
$$

SAR (Source to Artifact Ratio) calculates the effects of artifact errors in the predicted sound output. Artifacts include distortions of the sources, or ”burbling” artifacts. Higher SAR values indicate the effects of artifact errors are minimal.

$$
SAR := 10 \log_{10} \frac{\left\|s_{target} + e_{interf} + e_{noise}\right\|^2}{\left\|e_artif\right\|^2} \tag{7}
$$

These metrics were used to evaluate how well our models and parameters were performing compared to the baseline model.

6 Results/Analysis

6.1 Parameter Tuning

Evaluation of the different models were done on a separate test set, which contained 50 slices of the recorded speech of length around 2 seconds that weren't part of the training set combined with 5 different noise sounds used in the training set.

The first comparison was made with two different parameters - number of layers (1-layer, 2-layers, 3-layers) and the usage of frequency weighted loss. Initial experiments proved that the binary masks performed better on the dataset compared to the soft masks, so all of the experiments were performed with the binary masks. Figure 6 shows the performance of the different models on the metrics sar_clean, sar_noise, sdr_clean, sdr_noise, sir_clean, sir_noise. The results show that the 2-layers model with frequency weighted loss performs the best in the test set.

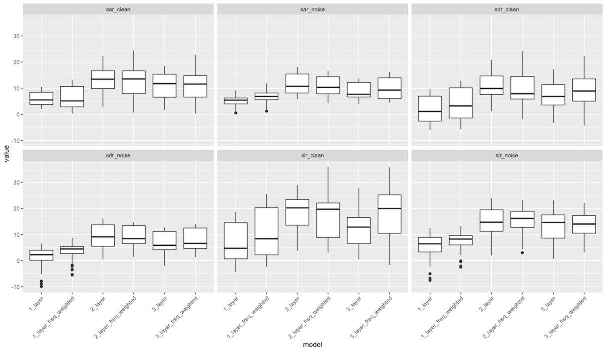

Figure 6: Metrics for different models

After selecting the parameters (2-layers, frequency weighted loss), we used the same parameters to train a bidirectional RNN. Figure 7 compares the performance of the bidirectional rnn model with the single directional rnn model, using the same parameters of 2-layers and frequency weighted loss. The bidirectional rnn slightly outperforms the single directional rnn in the metrics sir_clean and sdr_clean but in general performed worse. This could be due to the dataset not being large enough or the the training time not being long enough, as was the case when the 3-layers model didn't perform better than the 2-layers model in 6.

An example of the separated audio using the single directional 2-layers rnn model with frequency weighted loss could be found here (random example not in the training set without cherry picking):

Clean sound before mix: https://www.dropbox.com/s/thetxqddw9sc7ua/original_clean.wav?dl=0

Noise sound before mix: https://www.dropbox.com/s/9175crxcmzwxf6l/original_noise.wav?dl=0

Combined audio: https://www.dropbox.com/s/f0eoii5bl2sv6hd/combined.wav?dl=0

Separated clean sound: https://www.dropbox.com/s/fmk0alq50qjlspi/output_clean.wav?dl=0

Separated noise sound: https://www.dropbox.com/s/6e7gnab5dmqua2v/output_noise.wav?dl=0

As a reference to how the sound compares to the metrics used, the metrics for this single case were:
sdr_clean: 14.1878
sdr_noise: 14.5744
sir_clean: 24.2048
sir_noise: 21.7441
sar_clean: 14.6600
sar_noise: 15.5286

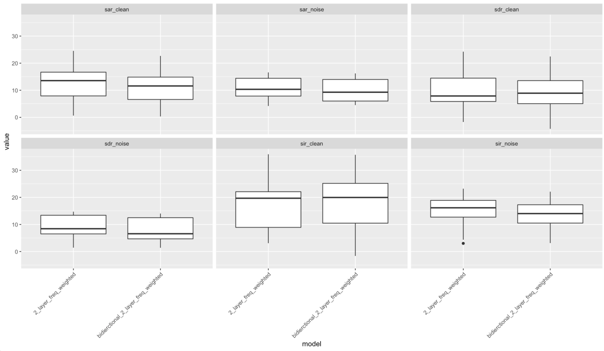

Figure 7: Metrics for regular / bidirectional rnn

6.2 Masking Iteration

After training the models, the predicted clean signal was added back to the combined audio to create a new output and calculate a new binary mask. Contrary to the expectation that the amplification of the estimated clean signal would help the model identify the clean signal better, all of the metrics dropped after a short increase in the first few iterations, as shown in Figure 8. Figure 9 describes the convergence of the mask at each iteration, plotting the average number of changes in the masks per iteration (Frobenius norm of old_mask - new_mask). The value of iteration 1 is 0 since there is no previous mask to compare. As expected, the changes in mask decreases through iteration, though we stopped running the iterations until convergence due to the performance drop.

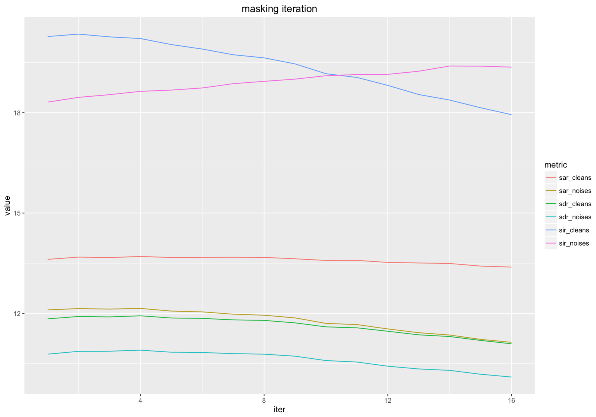

Figure 8: Metric performance for masking iteration

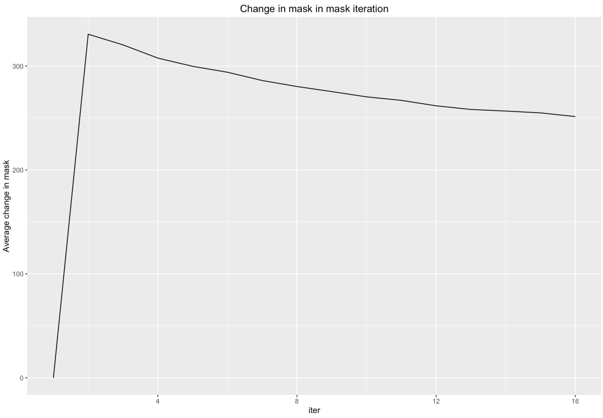

Figure 9: Convergence of masking

We believe the drop in performance was due to the model being only trained on a 50/50 mix of the clean and noise sounds, so that the model had a harder time if the noise signal was harder to differentiate from the clean signal. However, when repeating the same experiment with a test set containing a 75/25 mix of the clean and noise sounds, we observed that the initial results were better for the metrics for the clean signal (sdr_clean, sar_clean, sir_clean) and as expected, worse for the metrics for the noise signal(sdr_noise, sar_noise, sir_noise). As seen in Figure 10, the initial performance of the model on the test set with a larger mix of the clean signal starts with sir_clean over 20 dB, compared to slightly below 20 dB in the 50/50 mix as shown in Figure 8. Thus we estimate that the reason for the masking iteration not working is due to amplification of the noise signals that are added from the estimated clean signal confusing the model further.

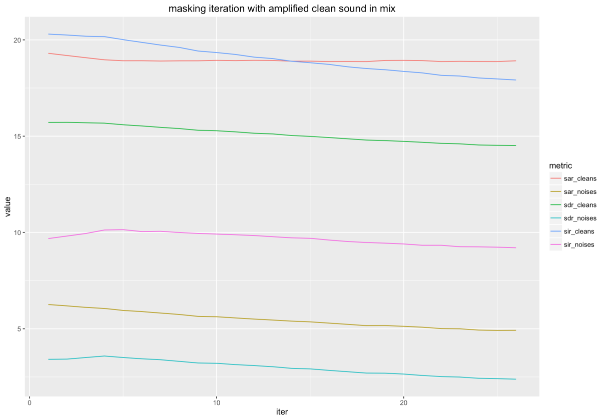

Figure 10: Metric performance for masking iteration when clean signal is amplified

7 Conclusion/Future Works

The simple RNN model works surprisingly well with a short training time (the 2-layer frequency weighted loss model reached near convergence after around 2 hours of training). Compared to our baseline model, an one-directional single-layer neural network model, our implemented model achieved gains in 15dB SIR, 7-8dB SAR/SDR for the clean signals. This shows that the model proposed by this paper effectively provides answers to many of the limitations that existed in the baseline model.

As future steps, it would be significant to look more at the drop in performance when amplifying the estimated clean signal in the combined audio. Our initial intuition was that amplifying the clean signals would create more accurate masks, and it may be meaningful to test whether our intuition was misguided or the model is not sufficiently robust. In addition, as masking separates clean and noise sounds based on frequency bins, it is possible that the model may face difficulties separating sounds with similar frequency ranges. In real life, this would be applicable to trying to separate out a main speaker talking in a crowded environment where numerous people are talking in the background. As our model focused on data obtained from DAPS dataset, the next step to take would be to test and improve the model to run with unknown multiple sound sources.

Acknowledgments

Thanks to Andrew Maas and the teaching staff of CS224S at Stanford University for introducing the material and providing the computing resources on Google Cloud Platform to train the models.

References

Emad M. Grais, Mehmet Umut Sen, and Hakan Erdogan. 2013. Deep neural networks for single channel source separation. Faculty of Engineering and Natural Sciences, Sabanci University. https://arxiv.org/pdf/1311.2746.pdf.

Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. 2014. Deep learning for monaural speech separation. Adobe Research. http://www.smaragd.is/pubs/huang-icassp2014.pdf.

Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. 2015. Joint optimization of masks and deep recurrent neural networks for monaural source separation. In *IEEE Transactions on Audio, Speech and Language Processing*. IEEE, pages 1–12. https://arxiv.org/pdf/1311.2746.pdf.

Tae Gyoon Kang, Kisoo Kwon, Jong Won Shin, and Nam Soo Kim. 2015. Nmf-based target source separation using deep neural network. In *IEEE Signal Processing Letters*. IEEE, pages 229–233. https://doi.org/10.1109/LSP.2014.2354456.

Andrew L. Maas, Quoc V. Le, Tyler M. ONeil, Oriol Vinyals, Patrick Nguyen, and Andrew Y. Ng. 2012. Recurrent neural networks for noise reduction in robust asr. In *INTERSPEECH*. https://research.google.com/pubs/pub45168.html.

Emmanuel Vincent, Remi Gribonval, and Cedric Fevotte. 2010. Performance measurement in blind audio source separation. In *IEEE Transactions on Audio, Speech and Language Processing*. Institute of Electrical and Electronics Engineers, pages 1462–1469. https://hal.inria.fr/inria-00544230.

A Supplemental Material

The source code for this project can be found at the following github repository: `https://github.com/hjkwon0609/speech_separation`