Speech Denoising Neural Networks Based on Mel-spectrogram and STFT features

C54076055 于治維 **Jhih-Wei Yu, C44071053** 卓庭萱 **Ting-Hsuan Cho,**

F94076102 林雨農 **Yu-Nong Lin, F94071021** 張荷昕 **Ho-hsin Chang**

Anonymous submission

Paper ID

1. Introduction

Reconstructing clean speech from noisy ones is crucial for many real-world applications in speech technology, such as automatic speech recognition and hearing aids. Upon seeing the potential of applying machine learning techniques in the field of audio noise reduction, the industrial technology research institute(ITRI) held a competition on AIdea, an artificial intelligence collaboration platform, expecting to identified and purify human voices from noise.

ITRI provided a dataset containing 42828 sets of recording data with noise and corresponding ground truth for training. It contains various kinds of noise like, children laughing, caf music, dog bark, etc. Also, the dataset has contained speech in various languages with different accents. So, this makes it harder for preprocessing since different language family has their own characteristic and suitable feature extraction technique.

1.1. Previous studies

Speech denoising aims to improve the quality of distorted speech signals caused by background noises, interference and recording devices.Traditional speech denoising models are largely based on statistical results of signals. Some common approaches including spectral subtraction, Wiener filtering, and minimum mean square error (MMSE) apply filters to separate noise components from clean speech components in noisy speech signals.

Another type of SE algorithms performs linear transformation in order to separate noise and clean speech subspaces, such as singular value decomposition (SVD) and principal component analysis (PCA).

However, traditional SE algorithms only perform well under certain stationary noise condition and with seen data. In this study, we will apply deep learning-based SE methods for better nonlinear feature mapping ability. Autoencoders have been commonly used for robust feature selection and extraction in image processing and other applications.

2. System framework

2.1. Pre-processing

The short-term Fourier transform (STFT) process is to divide a long-term signal into several shorter signals of equal length, and then separately calculate the Fourier transform of each shorter segment. The method implemented in Python’s scipy will calculate the amplitude and phase of each frequency band in a window and represent it as a complex number. We decompose this number into real and imaginary matrices (spectrograms) as inputs and output.

2.1.1 Spectrogram

In order to make it easier for the model to process the data and retain as much information as possible while reducing the file size, we converted the FLAC audio into a spectrum map (frequency axis resolution 256) using a window of size 512, and then cut it into 256*256 size images by taking 256 data points on the time axis. We have tried two types of spectrogram as follows.

2.1.2 Mel-Frequency Cepstral Coefficients

Mel-Frequency Cepstral Coefficients (MFCC) are a set of coefficients used to create a Mel cepstral. From the segments in the music signal, we can obtain a set of cepstrums that are sufficient to represent the music signal, and the Mel cepstral coefficients are the cepstrums derived from the spectrum of the spectrum.

The biggest feature of the Mel cepstrum is that the frequency band on the Mel cepstrum is evenly distributed on

the Mel scale. Compare with the general linear cepstrum, the representation form of Mel cepstrum is closer to the human nonlinear auditory system. Therefore, Mel cepstrum is often used for processing in the technology of audio compression.

For speech or speaker recognition, the most commonly used acoustic features are mel-scale frequency cepstral coefficient (MFCC for short). MFCC takes human perception sensitivity with respect to frequencies into consideration, and therefore are best for speech or speaker recognition.

The computation process of MFCC is shown in the figure 1.

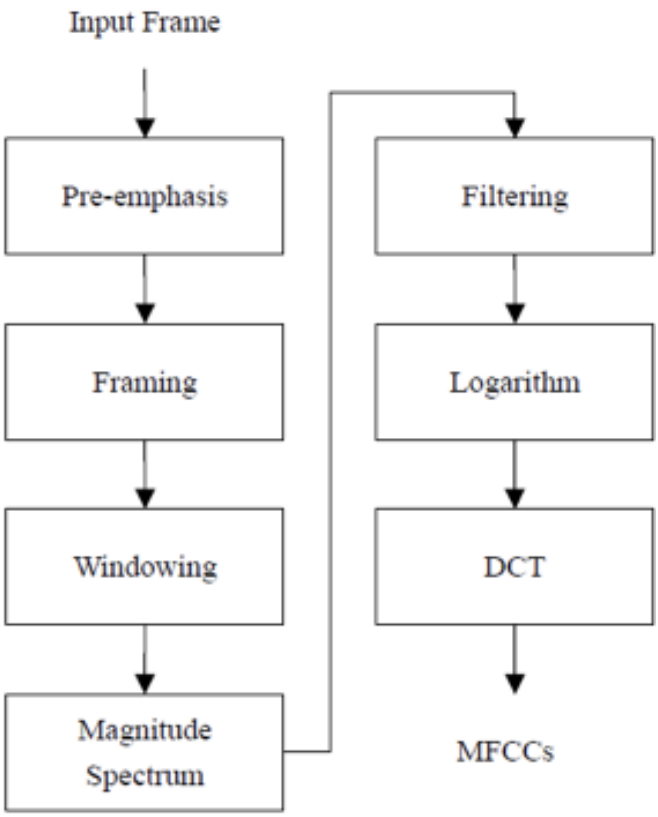

Figure 1. The computation process of MFCC.

After trying various models, we found that the spectrogram produced by ResUNet had the best result (smallest MSE), but then we realized that the phase information wasn't involved in the mel- spectrogram, which caused the output distorted. So we tried the other method.

2.1.3 Short-term Fourier transform (STFT)

Short Time Fourier Transform is a variation of Fourier Transform. The process of calculating Short Time Fourier Transform (STFT) is to divide the long time signal into several shorter equal length signals, and then calculate the Fourier Transform of each shorter segment separately. It is usually used to depict the changes in the frequency and time domains, and is one of the important tools in time-frequency analysis.

A function can be multiplied by a window function that is non-zero only for a period of time before performing a one-dimensional Fourier transformation. The window function is then shifted along the time axis, and the resulting series of Fourier transformations are lined up to form a two-dimensional representation. Mathematically, this operation can be written as follows

$$X(t,f) = \int_{-\infty}^{\infty} w(t-\tau)x(\tau)e^{-j2\pi f\tau}\,d\tau$$

In the case of discrete time, the data will be cut into several large number of frames, and each group of frames will usually overlap with each other to avoid the error of the boundary caused by the cutting method. And each group of frames will be Fourier transformed in each group of the complex results will be summed up again to get the magnitude and phase of time and frequency change at each point. Mathematically, this operation can be written as follows

$$\mathbf{STFT}\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

$$\mathbf{STFT}\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n}$$

Similarly, where $w[n]$is the window function and $x[n]$ is the signal to be transformed. Although different from the mel spectrum in physical meaning, in structure, the STFT spectrogram is similar to the MEL spectrogram, so we expect that ResUNet can handle STFT spectrum well while retaining the phase information.

2.1.4 Copy Padding

When a piece of audio is cut to the end, there may be less than 256 data points left, we will copy this piece until the 256 data points are made up.

2.1.5 Normalize

To make the input data better processed by the model, we need to normalize the absolute value of the input to between 0 and 1. At the same time, we also consider that sign of data contains important information, so we need to zoom the spectrum to 0 as the origin (instead of the mean). Due to the property of human perception, we considered the following transformations after normalize.

2.1.6 μ -law

$$F(x) = sgn(x)\frac{\ln(1+\mu|x|)}{\ln(1+\mu)}, \quad -1 \le x \le 1$$

$F^{-1}(y) = sgn(y)\frac{(1+\mu)^{|y|}-1}{\mu}, \quad -1 \le y \le 1$ The above equations are the transformation function and its inverse function, where μ =255, x is the origin signal, and y is the transformed signal.

2.2. Models

2.2.1 AutoEncoder

The first model is an autoencoder. Its most classic feature is the learning framework composed of the encoder and the decoder. The structure of the model has a total of 5 layers. The 256*256 spectrum of the previously processed audio is used as the input of the model. The steps that each layer will go through are convolution, ReLU and batch normalization. The size of each layer after convolution is shown in the figure 2. After 5 layers of convolution, it enters the decoder part, and it will reverse deconvolution according to the steps of convolution and the number of layers until it finally returns to the original 256*256 size, that is, the spectrum after denoising learning is completed.

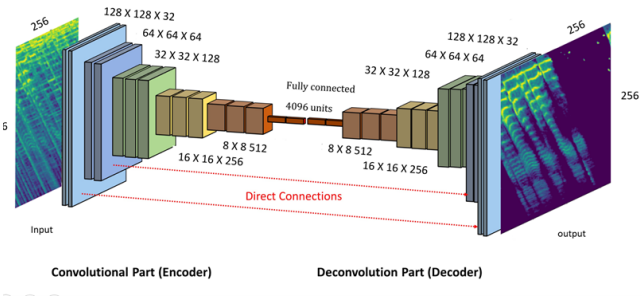

Figure 2. The structure of AutoEncoder.

2.2.2 U-Net

The second model is UNET 3. The main difference from the previous autoencoder is that the steps experienced in each layer become convolution, leakyReLU, convolution, leakyReLU, maxpooling and entering the next layer. Except that each layer undergoes two convolutions, the difference between using leakyReLU function and ReLU function is that leakyReLU retains negative values to avoid the vanishing gradient problem. Vanishing gradient problem means that in each training iteration, the updated value of the neural network weights is proportional to the partial derivative of the error function. However, in some cases, the gradient value can almost disappear, so that the weights cannot be updated effectively, and the neural network may not be able to continue training at all. Finally, the decoder part will complete the result of the 5-layer convolution, and reverse the layer-by-layer up sampling to output the final result. Between the layers of the same size of the encoder and the decoder is concatenate function. It's used for feature fusion, which is to connect the vector features of the layer directly with the same size layer in the decoder.

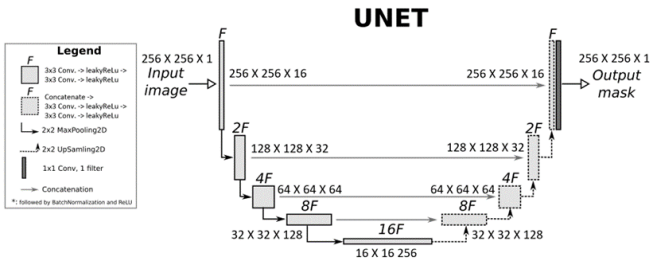

Figure 3. The structure of U-Net.

2.2.3 ResUnet

Our ResUnet is a adaptation of the traditional 4 layer-Unet structure with additional residual connection. In encoder part 4, each basic unit in encoder path is composed of a residual block and a convolution block. The output of residual block will be half of its input width and height with doubled channel. In convolution block, first, the output of last down sampling layer is batch normalized, and after two 2D-convolution layers, the shape of feature map remains the same. Finally, the output of each basic unit in encoder path is the pixel-by-pixel addition of these the residual and convolution blocks.

In decoder part 5, the up sampling layer is basically the reverse process of the encoder down sampling above. The residual block in decoder will decrease its channel size to match with the shape of corresponding layer in encoder path. And the convolution block will not change its shape after two 2D-convolution layers. In short, the input of each layer in decoder path will be up sampled first, then concatenate with the layer which shares the same width and height. Afterwards, the output of this layer will be the new input of the residual and convolution block. Finally, the output of each basic unit in decoder path will be the pixel-by-pixel addition of the residual and convolution blocks mentioned above.

2.3. Losses and Metrics

Since our target is a regression problem, we choose to apply Mean Absolute Error(MAE) and Mean Squared Error(MSE). However, we discovered that the loss curve 6 tends to fluctuate even after the loss has converged. The loss drops to approximately zero within the first 5 epochs and

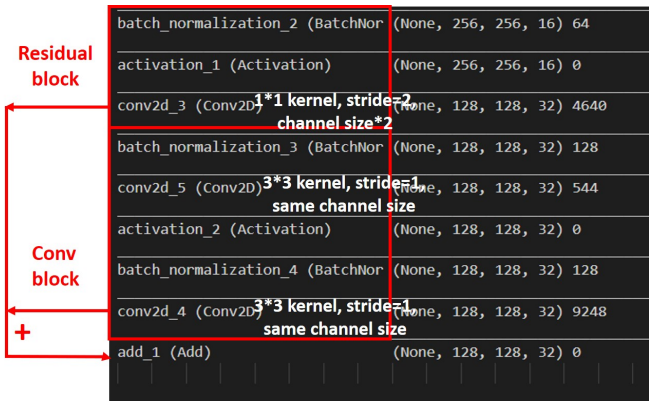

Figure 4. Composition of layers of each basic unit in encoder path

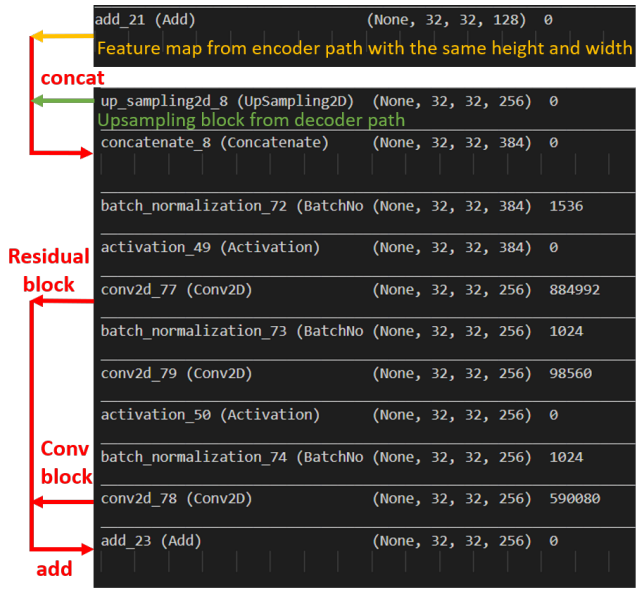

Figure 5. Composition of layers of each basic unit in decoder path

fluctuates around the same value until 100 epochs. Therefore, after multiplying MSE by the weighting of one million 7, the loss decreases to a greater extent and converges more steadily without up and down fluctuations. The multiplied MSE will be used as loss function throughout the rest of our experiment.

3. Results

3.1. Mel-spectrogram

The models proposed above are trained under the same condition where batch size equals to eight and Adam optimizer(learning rate =0.01, decay=0.01). From 14, we can observe that ResUNet has the most converged loss curve among autoencoder, UNet and ResUNet after training for 10 epochs. Therefore, we trained ResUNet under the same

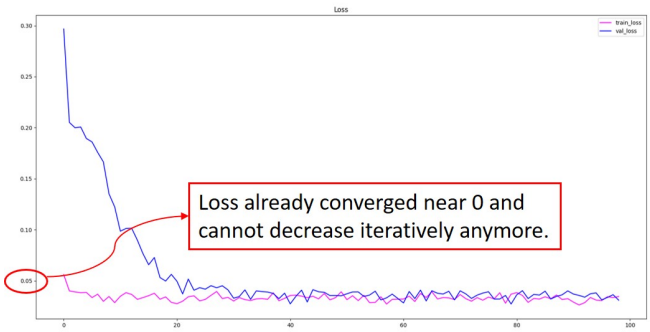

Figure 6. Loss curve prior to multiplying the weighting

$$MAE = \frac{|(y_i - y_p)|}{n}$$

$$RMSE = \sqrt{\frac{\sum (y_i - y_p)^2}{n}} * 1000000$$

y_i = actual value
y_p = predicted value
n = number of observations/rows

Figure 7. Loss function

condition for 50 epochs. We will evaluate the model's ability to denoise based on the reconstruction quality of the mel-spectrogram.

The reconstruction result of Autoencoder 10 shows that the global features have been learned but the detailed features may need a few more training epochs to be learned. The reconstruction result of UNet 11 after 10 epochs are still very unclear. Thus, it shows that UNet might not be the most suitable network structure for our application. The reconstruction result of ResUNet 12 after 10 epochs show promising progress since it not only captured most of the global features, but also successfully reconstructed some of the local features comparing with the expected output 9.

From 10, 11, and 12, ResUNet ranked first in reconstruction quality, while autoenoder ranked second and UNet ranked third. The results of ResUNet after training for 50 epochs comparing to 10 epochs, the former has more complete reconstruction in the high frequency component.

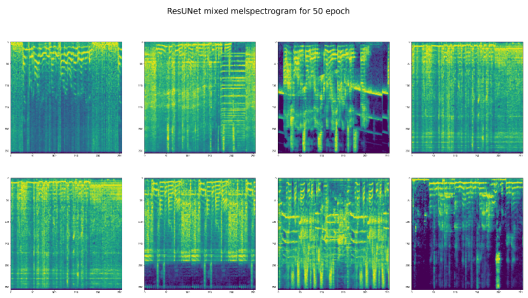

Figure 8. Reconstructed image of noisy input

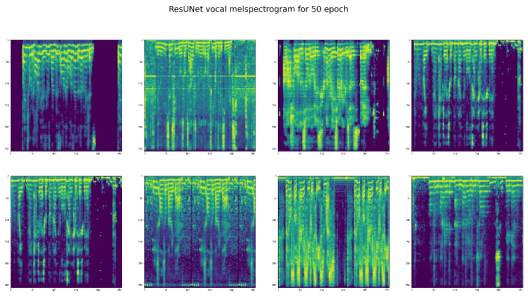

Figure 9. Reconstructed image of clean input

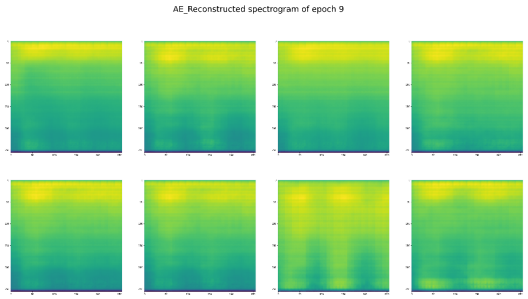

Figure 10. Reconstructed spectrogram of 10 epoch of Autoencoder

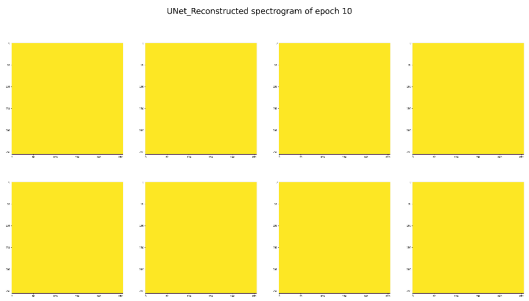

Figure 11. Reconstructed spectrogram of 10 epoch of UNet

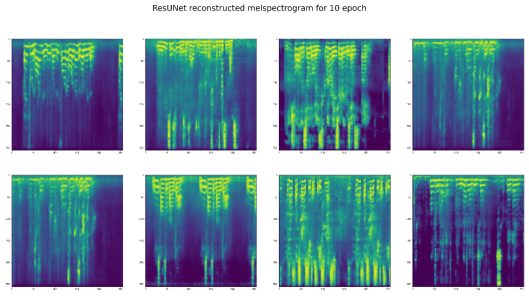

Figure 12. Reconstructed spectrogram of 10 epoch of ResUNet

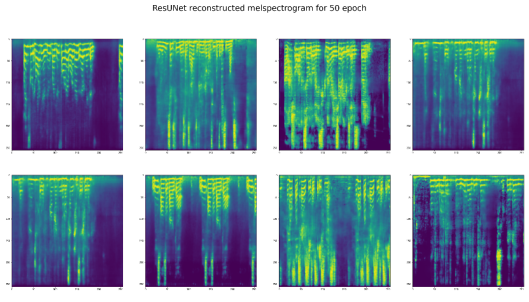

Figure 13. Reconstructed spectrogram of 50 epoch of ResUNet

3.2. STFT

In this section, we use STFT as the input feature. The reason why we use STFT is that logarithm of the mel-spectrogram during data pre-processing only kept the positive part of the signal, resulting in the loss of negative information that could not be restored. Nevertheless, our goal is to minimize the loss between spectrogram to audio transformation because the information loss are unrelated to the actual model reconstruction performance. At the same time, we want to keep the characteristic of logarithm operation which makes the features more visible. Finally, we came up with a method to satisfy both sides of the equation, which is to use the STFT feature composed of both real and imaginary parts, and then multipled by mu law conversion function. In this case, the information can be retained while keeping the advantages of signal enhancement effect of logarithmic operation.

4. Conclusion

To summarize our work, we have made several attempts in modifying the network architecture and feature type to optimize the speech denoise network and minimize the information loss of spectrogram-to-audio conversion. Based on the preliminary results of speech reconstruction, we concluded that ResUNet is the most suited network architecture

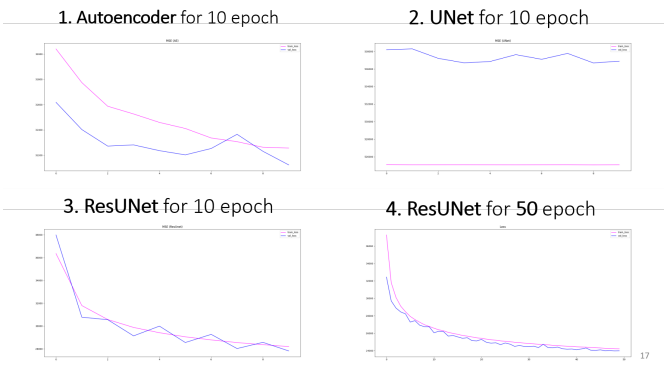

Figure 14. Loss curve of the 4 condition

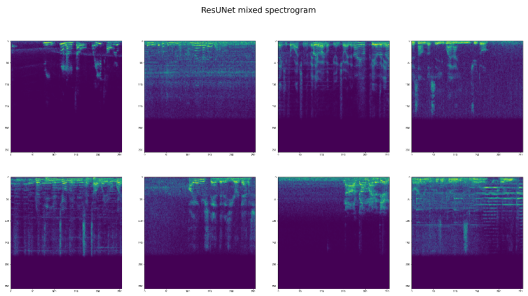

Figure 15. Reconstructed STFT of noisy input

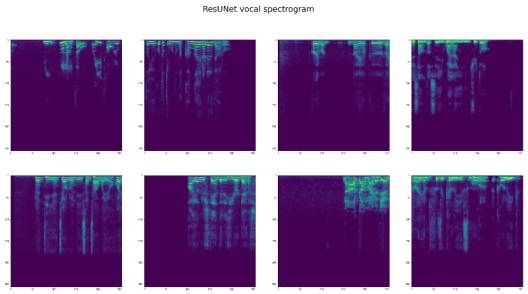

Figure 16. Reconstructed STFT of clean input

for speech denoise, and STFT being a more suitable feature for minimizng the information loss.

↓ **Check this link and listen to our reconstructed audio !**

References

AIdea人聲語音去噪競賽 Link

Lu, X., Tsao, Y., Matsuda, S., Hori, C. (2013, August). Speech enhancement based on deep denoising autoencoder. In Interspeech (Vol. 2013, pp. 436-440). Link

Github page for Speech-enhancement Github

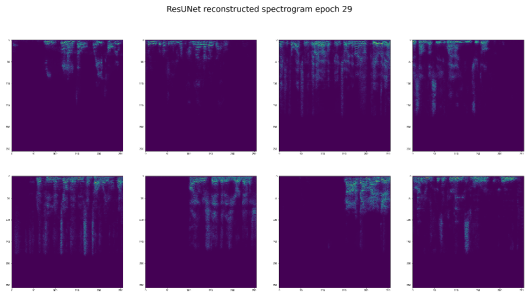

Figure 17. Reconstructed STFT of 30 epoch of ResUNet

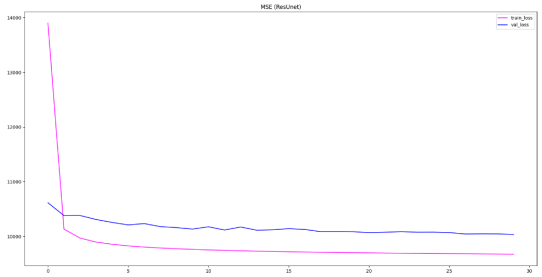

Figure 18. Loss curve of STFT of 30 epoch of ResUNet

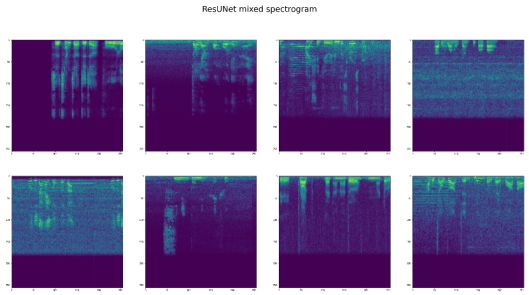

Figure 19. Reconstructed STFT of noisy input

Chuang, Fu-Kai, et al. ”Speaker-Aware Deep Denoising Autoencoder with Embedded Speaker Identity for Speech Enhancement.” Interspeech. 2019.

Grais, Emad M., and Mark D. Plumbley. ”Single channel audio source separation using convolutional denoising autoencoders.” 2017 IEEE global conference on signal and information processing (GlobalSIP). IEEE, 2017.

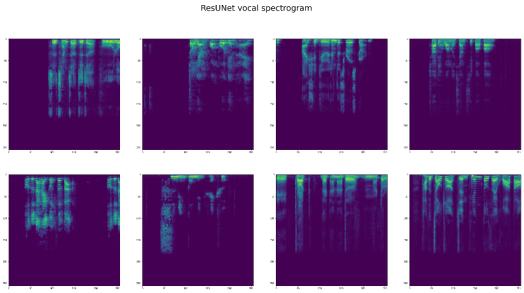

Figure 20. Reconstructed STFT of clean input

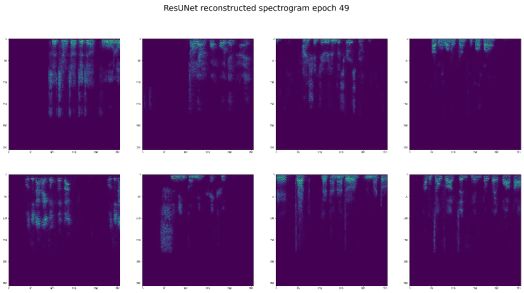

Figure 21. Reconstructed STFT of 30 epoch of ResUNet

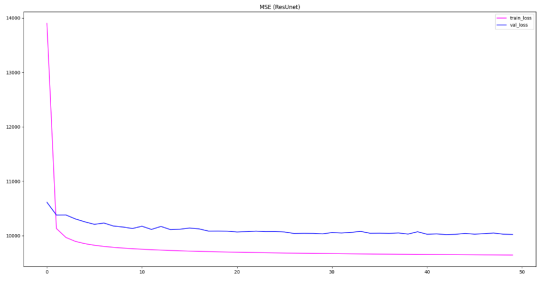

Figure 22. Loss curve of STFT of 50 epoch of ResUNet