Novel Deep Learning Approaches for Single-Channel Speech Enhancement

Kai Wang

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Master of Applied Science (Electrical and Computer Engineering) at

Concordia University

Montréal, Québec, Canada

July 2022

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Kai Wang

Entitled: Novel Deep Learning Approaches for Single-Channel Speech Enhancement

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Electrical and Computer Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. Hassan Rivaz

_____ Examiner
Dr. Arash Mohammadi

_____ Examiner
Dr. Hassan Rivaz

_____ Thesis Supervisor(s)
Dr. Wei-Ping Zhu

_____ Thesis Supervisor(s)

_____ Thesis Supervisor(s)

Approved by _____
Dr. Yousef Shayan, Chair
Department of Electrical and Computer Engineering

_____
Dr. Mourad Debbabi, Dean
Faculty of Engineering and Computer Science

Date _____

Abstract

Novel deep learning approaches for single-channel speech enhancement

Kai Wang

Acquiring speech signal in real-world environment is always accompanied by various ambient noises, which results in the degraded intelligibility and quality of the speech. To make listeners hear the high-quality speech signal, speech enhancement is thus necessary, which aims to estimate the clean speech from noisy mixture by removing the background noises. Recently, deep-learning approaches, as a powerful tool, have impressively promoted the advancement of speech enhancement, which usually trains a neural network to learn the mapping function from the noisy speech to the clean speech with supervised learning. However, commonly used neural networks cannot efficiently leverage contextual information of speech signal while involving large footprint. For example, convolutional neural networks require to be deep enough for capturing sufficient receptive field due to the intrinsic locality. Besides, the recurrent neural networks cannot learn the long-term dependency of long speech sequences and moreover are computational-expensive because of the sequential processing.

To learn the contextual information of long-range speech sequencies, in the first contribution of this thesis, two novel attention-based transformer neural networks are proposed for single-channel speech enhancement: two-stage transformer and cross-parallel transformer neural networks, termed as TSTNN and CPTNN, respectively. The proposed TSTNN adopts multiple pairs of local and global transformer with cascaded connection to successively extract both local information of individual frames and global information across various frames, generating the contextual feature representation. To overcome the information leakage caused by shifting from one path to another in cascaded structure, the proposed CPTNN employs cross-parallel transformer blocks to extract local and global information in parallel, whose outputs are then adaptively fused by inner cross-attention mechanism for the generation of contextual information. Both proposed architectures incorporate the transformer blocks between convolutional encoder and decoder, where the outputs of transformer blocks are fed into a masking module to create a mask for filtering the encoder outputs which will be transformed to enhanced speech via decoder.

Extensive experiments indicate that the proposed TSTNN and CPTNN achieve a competitive or even superior performance compared to existing state-of-the-art methods in all evaluation metrics while having fewest trainable parameters.

While many competitive performances are realized by networks involving attention mechanism such as transformer as a part of them, it is worth-studying whether the attention is indispensable for resolving long-term dependency problem in speech enhancement, especially when considering the trade-off between computational efficiency and denoising performance. In the second contribution of this thesis, we propose an attention-free architecture based on multi-layer perceptrons (MLPs) for speech enhancement, named SE-Mixer, which consists of an encoder, a decoder and multiple mixer blocks in between. The mixer block is designed for efficiently extracting contextual information of long-range speech sequences. It employs temporal MLP in conjunction convolution and frequency MLP to iteratively capture multi-scale temporal information from various time scales and extract frequency information within each time step. Our experimental results demonstrate that the proposed SE-Mixer has a competitive performance and relatively low parameters compared to existing methods and without attention mechanism incorporated. We have also shown that the architecture without attention algorithm is likely to reach approximately the same effectiveness as compared with attention-assisted models but with significantly lower computational complexity.

Related Publications

The work of this thesis led to the following research publications:

1. **Kai Wang**, Bengbeng He, and Wei-Ping Zhu, "TSTNN: Two-stage transformer based neural network for speech enhancement in the time domain," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7098–7102.

2. **Kai Wang**, Bengbeng He, and Wei-Ping Zhu, "CPTNN: Cross-parallel transformer based neural network for time-domain speech enhancement," accepted by *IWAENC 2022 IEEE International Workshop on Acoustics Signal Enhancement (IWAENC)*.

3. **Kai Wang**, Bengbeng He, and Wei-Ping Zhu, "SE-Mixer: Towards an attention-free neural network for speech enhancement," submitted to *APSIPA 2022 IEEE Asia-Pacific Signal and Information Processing Association (APSIPA)*.

4. **Kai Wang**, Bengbeng He, and Wei-Ping Zhu, "CAUNet: Context-aware u-net for speech enhancement in time domain," in *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.

Acknowledgments

First and foremost, I would like to express my heartfelt gratitude and appreciation to my supervisor, Prof. Wei-Ping Zhu, for guiding me the academic research, supporting and encouraging me when I encountered difficulties, teaching me the theoretical knowledge and providing me valuable suggestions throughout my entire research.

I also would like to extend my deep gratefulness to my group members, Bengbeng He, Dr. Mojtaba Hasannezhad, Dr. Hongjiang Yu, Zhiheng Ouyang and all the laboratory members and friends for their nice help and comforts throughout my life during my master study.

Finally, I would like to express my deepest love to my parents. Their selfless love, encouragement and support are always a source of my strength to overcome all the obstacles and disappointments in my life.

Contents

List of Figures

List of Tables

List of Abbreviations

ASR	Automatic Speech Recognition
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
CV	Computer Vision
NLP	Natural Language Processing
STFT	Short-Time Fourier Transform
T-F	Time-Frequency
iSTFT	inverse Short-Time Fourier Transform
IBM	Ideal Binary Mask
IRM	Ideal Ratio Mask
cIRM	Complex Ideal Ratio Mask
ReLU	Rectified Linear Unit
FC	Fully-Connected
CED	Convolutional Encoder-Decoder
FCN	Fully Convolutional Network
LSTM	Long-Short Term Memory
GRU	Gated Recurrent Unit
CRN	Convolutional Recurrent Network
BN	Batch Normalization
ELUs	Exponential Linear Units
DCCRN	Deep Complex Convolution Recurrent Network
TCN	Temporal Convolutional Network
DNS	Deep Noise Suppression
PESQ	Perceptual Evaluation of Speech Quality

STOI Short-Time Objective Intelligibility

SSNR Segmental Signal-to-Noise Ratio

MOSs Composite Mean Opinion Scores

GELU Gaussian Error Linear Unit

PReLU Parametric Rectified Linear Unit

LN Layer Normalization

GN Group Normalization

MAE Mean Absolute Error

MSE Mean Square Error

SDR Signal Distortion Rate

MHA Multi-Head Attention

FFN Feed-Forward Network

PE Positional Encoding

TSTB Two-Stage Transformer Block

CPTB Cross-Parallel Transformer Block

TSTNN Two-Stage Transformer Neural Network

CPTNN Cross-Parallel Transformer Neural Network

DCCN Densely Connected Convolutional Network

APTB Average Parallel Transformer Block

PTB Parallel Transformer Block

MLP Multi-Layer Perceptron

SNR Signal-to-Noise

MACs Multiply-Accumulated Operation

Chapter 1
Introduction

Speech enhancement, as an indispensable front-end task of speech processing, aims to improve the perceptual quality and intelligibility of degraded speech by removing its background noise. It is widely adopted in many speech-related applications including voice communication, automatic speech recognition (ASR), hearing aids and audio systems. Researchers have explored speech enhancement based on statistical signal processing, leading to some popular traditional speech enhancement methods including Wiener filtering [1], spectral subtraction [2] and statistical model-based algorithms [3].

In recent years, deep learning emerges as a powerful tool to develop various data-driven approaches for solving traditional estimation problems with or without supervision. It is believed that the deep learning methods have significantly promoted the rapid advancement of speech enhancement. Some popular deep neural networks, including fully-connected neural network, convolutional neural networks (CNNs), recurrent neural network (RNNs) have been widely studied for speech processing tasks such as speech enhancement, speech separation and speech recognition. Benefited from the achievement of deep learning on the computer vision (CV) and natural language processing (NLP), some approaches derived from these two fields have inspired the works in speech processing. Speech signals are naturally one-dimensional sequences with strong temporal corrections among sequence samples, which can be processed by sequence models referred to the NLP tasks. In addition, the short-time Fourier transform (STFT) is usually implemented on one-dimensional speech waveform to obtain the image-like spectrogram with time horizontal axes and frequency vertical axes, which can readily adopt the popular methods from CV tasks. However, the speech processing tasks are essentially different from CV and NLP, and some specific approaches are required to be designed for the specific tasks.

This thesis investigates both attention-based transformer and attention-free neural networks for single-channel speech enhancement. To solve the long-range independency problem of speech sequences, we design specific neural network modules and model structure to extract sufficient contextual information for improving the performance of speech enhancement. The

other target of this thesis is to propose low footprint models for applications with limited computational resources. In the following, we will briefly introduce the existing works of speech enhancement and some constituent components which will be used in our methods.

1.1 Neural networks for speech enhancement

Deep neural network systems for speech enhancement can be categorized into Time-Frequency (T-F) domain and Time domain methods. In T-F domain SE models, the spectrogram features of noisy speech are obtained by short-term Fourier transform (STFT), which will be processed by the denoising systems to estimate the clean spectrogram. On the other head, the time domain methods directly estimate the clean speech from the noisy waveform.

1.1.1 Time-Frequency domain models for SE

T-F domain methods take the spectrogram features as the inputs since some harmonics of the speech signal exists in the spectrogram, which is extremely important for speech enhancement. By using STFT, the spectrogram representation of given noisy speech can be formulated as:

$$X(t,f) = S(t,f) + N(t,f) \tag{1}$$

where $X(t,f)$, $S(t,f)$ and $N(t,f)$ denote the spectrogram representation of noisy speech, clean speech and noise speech, respectively, t is the time frame, f means the frequency bins. We further represent the noisy spectrogram with Cartesian coordinate representation $\tilde{X} = \tilde{X}_{real} + j\tilde{X}_{img}$. Namely, the spectrogram $\tilde{X}$ is decoupled into real part $\tilde{X}_{real}$ and imaginary part $\tilde{X}_{img}$. Then, the magnitude and phase features can be written as:

$$\tilde{X}_{mag} = \sqrt{\tilde{X}_{real}{}^2 + \tilde{X}_{mag}{}^2} \tag{2}$$

$$\tilde{X}_{phase} = arctan2\,(\tilde{X}_{real}, \tilde{X}_{img}) \tag{3}$$

where $\tilde{X}_{mag}$ and $\tilde{X}_{phase}$ represent the magnitude and phase of noisy speech. To estimate the clean spectrogram from the noisy spectrogram, most of the T-F domain methods only enhance the noisy spectral magnitude to obtain the enhanced spectral magnitude, which will be

transformed into enhanced speech waveform by employing the phase information of noisy speech via inverse short-time Fourier transform (iSTFT), as shown in Fig. 1.1.

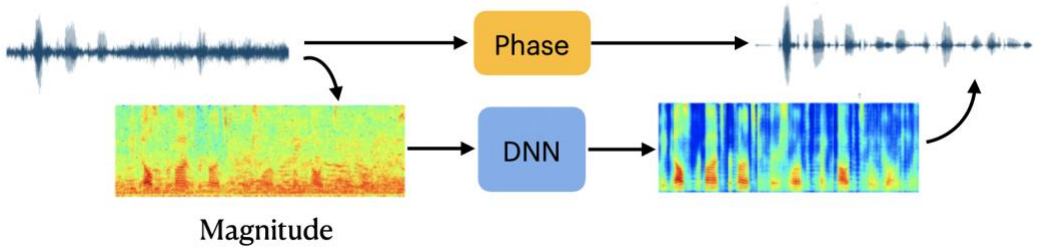


Figure 1.1: Basic framework for magnitude estimation in T-F domain methods [4]

To get the enhanced spectrogram, various output targets are investigated in different networks, including spectrogram-based targets and mask-based targets. The spectrogram-based targets are usually adopted by regression-based models which directly maps the noisy spectrogram into estimated clean spectrogram. In contrast, the mask-based targets are more commonly used in speech enhancement since these T-F masks contain some T-F elements which can filter out the noise components. More specifically, the T-F masks are obtained from denoising system and then multiplied with noisy speech spectrogram, generating the enhanced speech spectrogram. The denoising processing can be expressed as follows:

$$\tilde{S}(t,f) = X(t,f) \odot M(t,f) \tag{4}$$

where $\tilde{S}(t,f)$ and $X(t,f)$ are the enhanced speech spectrogram and noisy speech spectrogram, $\odot$ denotes the operation of element-wise multiplication, $M(t,f)$ represents the T-F mask. Various T-F masks are proposed for speech enhancement, including ideal binary mask (IBM) [5], ideal ratio mask (IRM) [6], complex ideal ratio mask (cIRM) [7] and so on. The IBM is a simple T-F mask to coarsely extract the clean speech from noisy speech, which can be formulated as follows:

$$IBM(t,f) = \begin{cases} 1, & if\ |S(t,f)|^2 - |N(t,f)|^2 > 0 \\ 0, & otherwise \end{cases} \tag{5}$$

where $IBM(t,f)$ denotes the T-F representation of IBM. There are two states 0 and 1 in each T-F unit of IBM mask. When the clean speech occupies the higher energy, the state will be 1 to retain the clean speech component. Inversely, the state 0 is taken to remove the noise speech component.

Compared with IBM, the IRM has more smooth value which is the ratio of clean speech spectrogram and noisy speech spectrogram, which is defined as follows:

$$IRM(t,f) = \left(\frac{|S(t,f)|^2}{|S(t,f)|^2+|N(t,f)|^2}\right)^{\beta} \quad (6)$$

where $IRM(t,f)$ denotes the T-F representation of IRM, β is scaling factor which is usually set as 0.5. The value of IRM is between 0 and 1, where the value closing to 1 represents the higher clean speech component within the T-F unit.

In most of T-F domain methods, the magnitude features are utilized as inputs and the phase features are retained for recovering the enhanced speech. However, the noisy phase information is usually ignored during denoising process, which restricts the performance of enhancing the speech quality [8]. To solve this problem, the cIRM is proposed to enhance the magnitude and phase information of noisy speech. Given the complex representation of clean speech S and noisy speech X, the cIRM can be formulated as follows:

$$cIRM = \frac{X_r S_r + X_i S_i}{{X_r}^2 + X^2} + i\frac{X_r S_i - X S_r}{{X_r}^2 + {X_i}^2} \quad (7)$$

where S_r and S_i represent the real and imaginary parts of the clean complex spectrogram, X_r and X_i represent the real and imaginary parts of the noisy complex spectrogram. By using the cIRM, the magnitude and phase of speech are processed during enhancing process.

FC-DNN models

The authors in [4] proposed a DNN-based model for speech enhancement with masking method as shown in Fig. 1.2. DNN is a simple and basic architecture of neural networks, which consists of an input layer, multiple hidden layers and an output layer. To provide more non-linear representation, the rectified linear unit (ReLU) [9] is inserted in each layer. The ReLU activation function will not activate neurons when receiving the negative inputs, which could bring more sparsity and non-linearity, and also avoid the gradient vanishing problem. In order to alleviate the overfitting problem caused by the deep hidden layers, the dropout operation is applied to deactivate some neurons with some proportion. The output layer is used to transform hidden features into expected output including enhanced speech or T-F mask in speech enhancement. For example, in Fig. 1.2, a DNN adopts several fully-connected (FC) layers to learn the

estimated mask which is multiplied with input speech to produce the enhanced speech. For cIRM estimation [6], the DNN outputs spectral masks for real and imaginary parts by employing two different output layers as shown in Fig. 1.3.

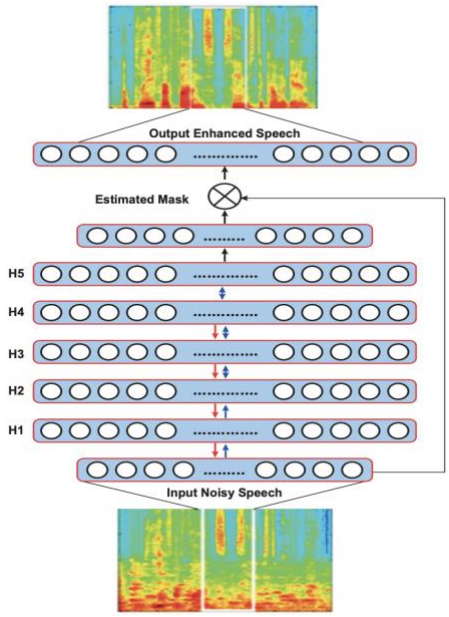


Figure 1.2: DNN for magnitude estimation [4]

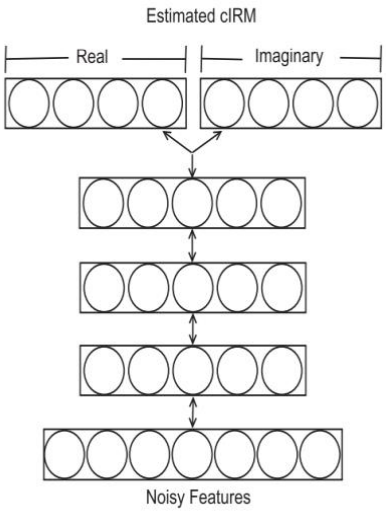


Figure 1.3: DNN for cIRM estimation [6]

CNN models

Different from DNN structures, CNNs typically contain less trainable parameters due to its local operation and weight sharing property. The CNNs are originally proposed for image recognition [10], where CNN layers are used to efficiently extract spatial features of input image, generating the high-level features to proceed the classification task by a simple classifier. Generally, as shown in Fig. 1.4, the CNN based structure receives the 2-D image as input and adopts multiple staked groups of a convolutional layer and a max-pooling layer to extract features of image, where the convolutional layer extracts the spatial features within a local region and the pooling layer performs the down-sampling to reduce the resolution of feature maps. After some CNN layers, the high-level features are obtained to pass through the FC layers to classify the given image.

In speech enhancement, the CNN structures are commonly adopted to process the 2-D spectrogram which is regarded as an image. As indicated in Fig. 1.5, the CNN takes the real and

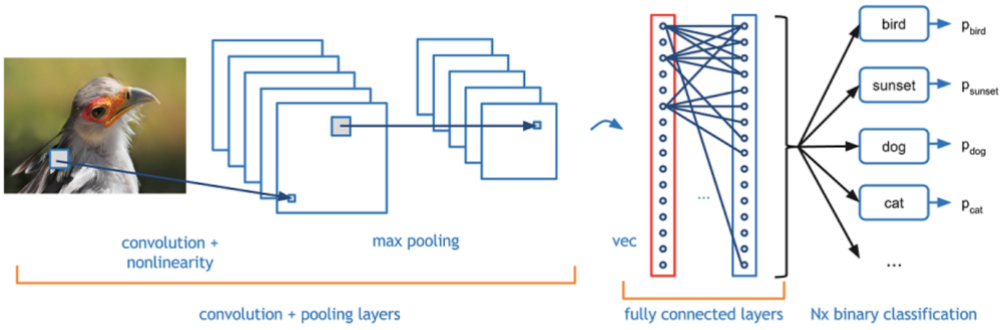


Figure 1.4: CNN for image classification [11]

imaginary (RI) spectrograms as the input, where the real and imaginary parts are packed together along the channel dimension, forming the 3-D input data. Two sub-layers are employed in the high-level features from CNN layers to estimate the real and imaginary spectrogram.

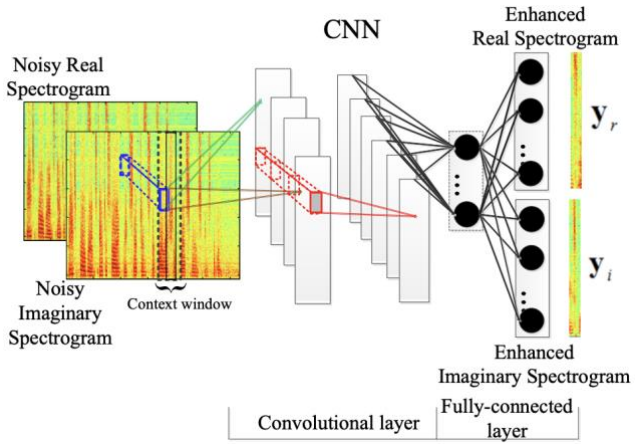


Figure 1.5: A CNN for the estimation of real and imaginary spectrogram [12]

The authors of [13] proposed a convolutional encoder-decoder (CED) network for speech denoising, which consists of symmetric encoder layers and decoder layers. Each encoder layer is comprised of a convolutional layer and max-pooling layer, and each decoder layer contains the convolutional layer and up-sampling layer. However, the max-pooling and up-sampling operations may bring potential leakage of information. Researchers further proposed a fully convolutional network (FCN) with encoder-decoder structure by removing the pooling and up-sampling layer, which directly maps the noisy magnitude spectrogram into enhanced magnitude spectrogram as shown in Fig. 1.6.

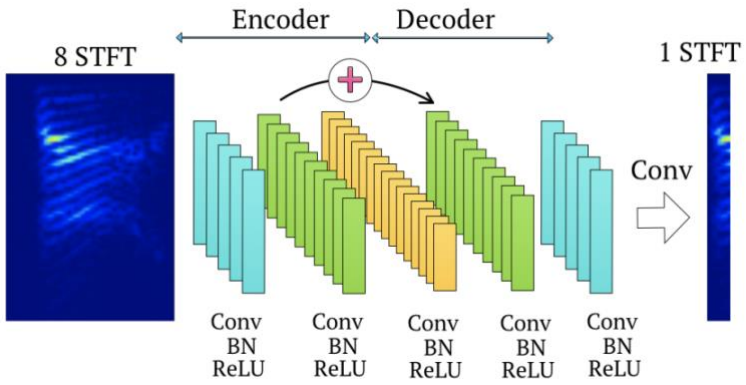


Figure 1.6: Convolutional encoder-decoder structure for magnitude estimation [13]

RNN models

However, CNNs require to be deep enough to capture high-level contextual representation due to the local operation of convolution. To enlarge the receptive field of features, some other layers are adopted including max-pooling layer or convolution with a shift size over one. Different from CNNs, the RNNs including long-short term memory (LSTM) [14] and gated recurrent unit (GRU) [15], can naturally capture long-term dependency of sequences due to the memory mechanism, which is a powerful tool for sequence modeling. Many researchers combine the CNNs and RNNs to construct a composite structure, named convolutional recurrent network (CRN), where CNN-based layers are used to extract low-level features and RNN-based layers further extract the contextual information of features. The CRN structures have shown an impressive performance in speech enhancement [16, 17].

In [17], the authors concatenate RNN layers with CED structure to form a composite system for speech enhancement as presented in Fig. 1.7. It inserts the LSTM layers between encoder and decoder to learn the temporal information of speech features. More specifically, the encoder takes the magnitude spectrogram as input to extract spatial features by using several pairs of a convolution, batch normalization (BN) [18] and exponential linear units (ELUs) [19]. The convolutional layer maintains the time dimension and halves the frequency dimension by using a stride of 2 along the frequency direction. The ELUs are an alternative of ReLU with a better generalization and convergency. Following or next to the encoder, two LSTM layers are adopted to capture the long-term contexts by memory cells, generating the contextual features. Next, the decoder is used to reconstruct the magnitude spectrogram by using multiple groups of a

deconvolution, BN and ELUs, where the deconvolution is applied to double the frequency dimension of feature. Finally, the softplus activation is added at the end of decoder to constrain the outputs to be always positive. To improve the information flow and avoid the gradient vanishing, the skip connections are used to concatenate each encoder layer into corresponding decoder layer.

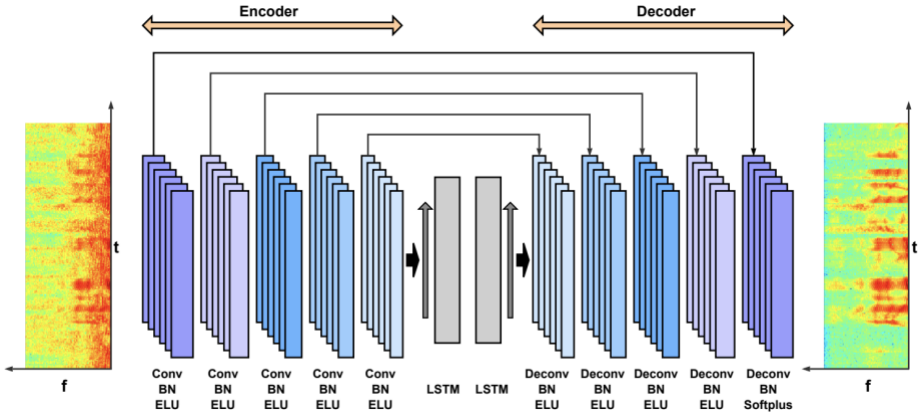


Figure 1.7: A composite model of CNN and LSTM for magnitude estimation [17]

Complex-valued models

For a long time, it is believed that the phase information is difficult to estimate. Therefore, some early works only focus on enhancing magnitude while ignoring the phase estimation, which fails to reach the upper bound performance of speech enhancement. Afterwards, some T-F domains methods utilize the RI spectrogram features as inputs or take the cIRM as the training target, which can roughly simultaneously consider magnitude and phase information during training. However, these models are still real-valued although the speech phase is taken into consideration. Recently, authors of [20] design a complex-valued neural network for speech enhancement, named deep complex convolution recurrent network (DCCRN). The DCCRN is built to receive the complex spectrogram, which proposes the complex-valued convolution and complex-valued LSTM to replace the traditional convolution and LSTM in CRN, respectively, as shown in Fig. 1.8. The encoder of DCCRN is comprised of complex encoder blocks, which include the complex 2-D convolution, complex-based BN and real-valued PReLU non-linearity [21] as shown in Fig. 1.9. In complex 2-D convolution operation, two traditional convolutional layers are adopted to parallelly process the real and imaginary part of complex spectrogram, which are

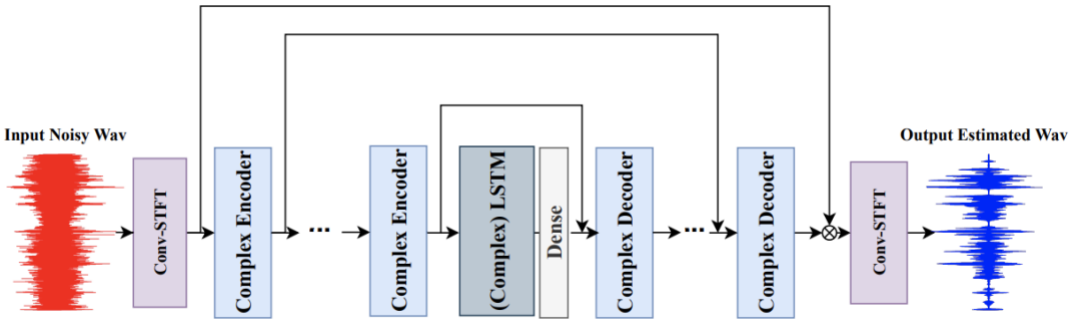
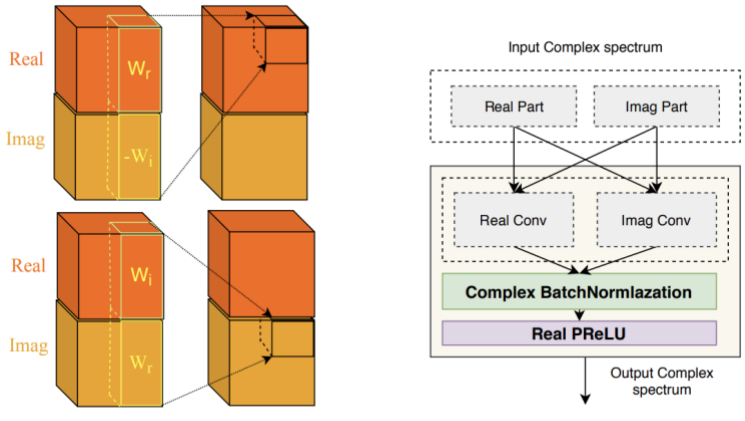


Figure 1.8: Structure of deep complex convolution recurrent network (DCCRN) [20]

named as real convolution and imaginary convolution. In addition, each convolution not only receives its relative inputs, but also proceeds the other part of spectrogram. For example, the real convolution simultaneously takes the real and imaginary spectrogram as inputs to learn their correlations, and the same procedure is taken by the imaginary convolution. Then, the outputs of real and imaginary convolutions are packed together to obtain the complex convolutional outputs, which will pass through the BN and PReLU non-linearity. Hence, each complex convolution includes four separated real-valued convolutions to learn the correlation between magnitude and phase via the simulation of complex multiplication. Similar to the complex convolution, the complex LSTM can be represented by replacing the real and imaginary convolution with real-valued real and imaginary LSTM. The complex decoder has the same structure as encoder except that the kernel size of decoder convolution is symmetric to that of encoder for recovering the feature dimension changed by encoder.



(a) complex convolution (b) complex encoder

Figure 1.9: Introduction of complex convolution and complex encoder [20]

1.1.2 Time domain models for SE

Different from T-F domain methods, the time-domain models directly estimate the clean speech waveform from noisy waveform, which can avoid the intractable phase estimation. Many existing time-domain SE systems adopt the U-Net architecture [22] to model the long-range speech waveform, where the convolutional encoder is used to extract high-level temporal features which are reconstructed by a symmetric decoder with skip connections from the encoder. The authors of [23] implements for the first time a Wave-U-Net structure for time-domain speech enhancement, which consists of 1-D convolutional encoder and decoder to perform the down-sampling and up-sampling as shown in Fig. 1.10. More specifically, the noisy speech waveform is first fed into multiple stacked down-sampling blocks to obtain the compressed features, where each block is comprised of a 1-D convolution and decimation operation to halve the time resolution. Then, the decoder takes the compressed feature as inputs to separate the clean and noise speech, which consists of multiple pairs of liner interpolation and convolution for up-sampling the time dimension with a factor of 2. To reuse the encoder information, a skip connection is used to link encoder layer and the corresponding decoder layer.

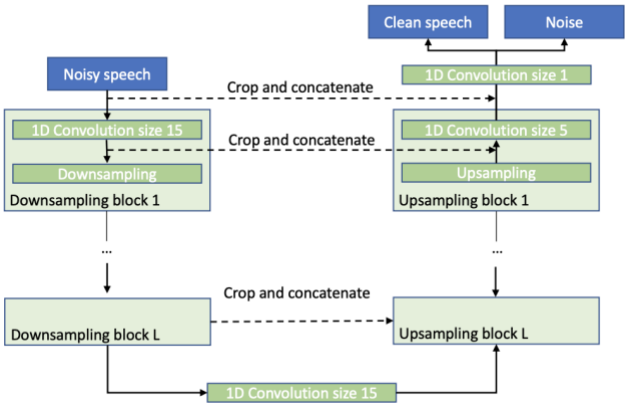


Figure 1.10: The architecture of Wave-U-Net [23]

The contextual information is important for speech enhancement, which can be achieved by enlarging receptive fields of CNNs. Common methods to this goal are to increase the network depth or enlarge the kernel size, which however brings huge computational burden. Authors of [24] proposed a dilated convolution to efficiently enlarge receptive field without adding parameters. Compared with conventional convolution, the dilated convolution can capture

features of larger region by inserting the dilation between kernel elements and maintaining the number of kernel parameters. For example, a 1×3 1-D conventional convolution operates on 1×3 regions each time as shown in Fig. 1.11 (a). However, as shown in the Fig 1.11 (b), the convolution with dilation 2 can extract information with 1×5 region each time while having the same number of parameters as the 1×3 1-D conventional convolution. As the dilated convolutional layer gets deeper, the dilation is exponentially increased to enlarge the receptive field.

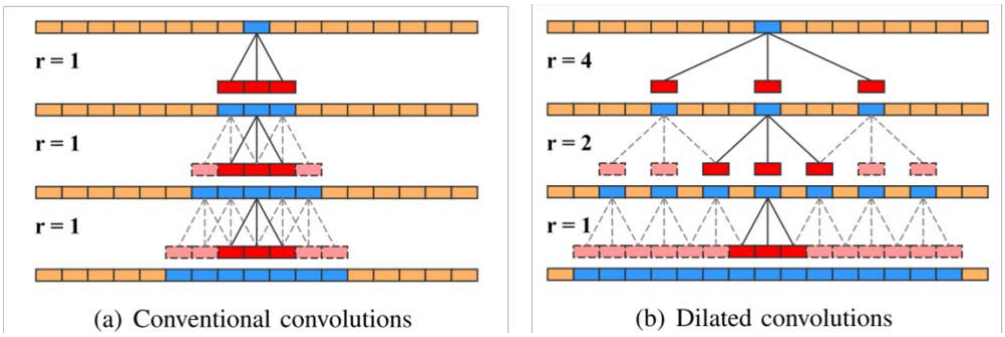


Figure 1.11: Comparison between convention and dilated convolutions [25]

Inspired by this study, the authors of [26] proposed a dilated convolution U-Net for speech enhancement in time domain. As shown in Fig. 1.12, the encoder is based on several convolutional blocks consisting of a conventional convolution to down-sample the dimension and a dense convolution block, which has multiple dilated convolutions with dense connection to increase the receptive fields and aggregate the information of former layers. After the encoder processing, the contextual speech features can be obtained, which will pass through the decoder for speech reconstruction. The decoder contains several groups of a sup-pixel convolution for up-sampling the dimension, a dense block and a concatenation from encoder layer.

Different from [25, 26], some researchers incorporate RNN layers between encoder and decoder to further extract the long-range dependency of features. In [27], the authors designed a composite model with LSTM and CED as shown in Fig. 1.13. The encoder layer includes two 1-D convolutions, of which one is followed by ReLU and the other one is followed by GLU non-linearity [28]. The first convolution is to decrease the temporal size of speech features and the second one is used to double the channel dimension. By a couple of encoder layers, the latent representations of speech waveform are obtained, which will pass through 2 LSTM layers for temporal extraction. The decoder layer contains a 1-D convolution for halving the channel

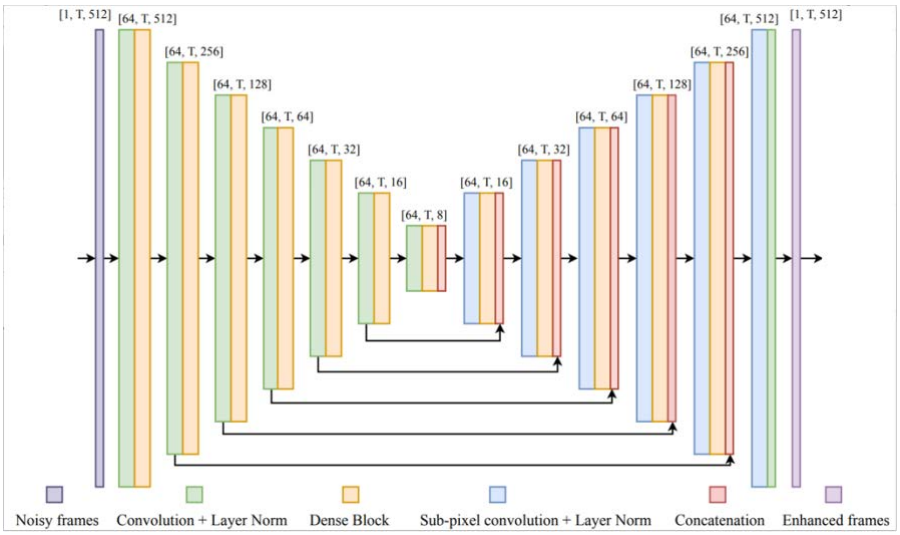


Figure 1.12: The structure of dilated convolution based U-Net for time domain SE [26]

dimension, a transposed convolution for increasing the temporal size, and a GLU activation in between. This model can be modified to be casual by using casual convolution and unidirectional LSTM.

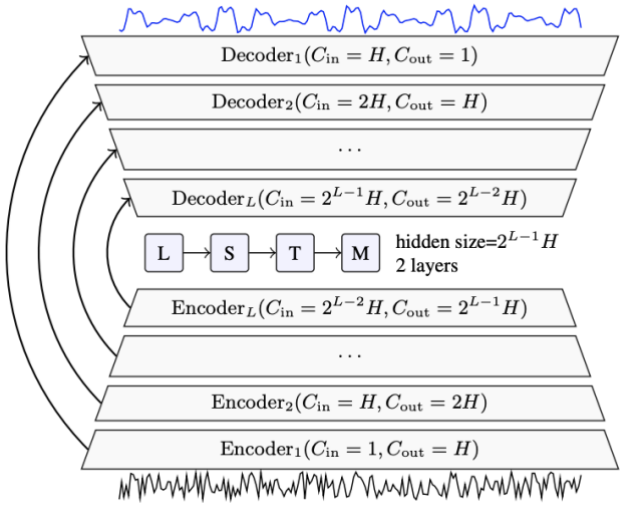


Figure 1.13: The overall structure of DEMUCS [28]

Considering that the RNNs only sequentially operate in sequence modeling, a temporal convolutional network (TCN) [29] is proposed as an alternative of RNNs to extract long-range dependencies from the past with a parallel processing. As indicated in Fig. 1.14, the proposed TCN-based SE system is realized in the time domain [30], which comprises a multilayer encoder,

a TCN separator and a multilayer decoder. The encoder adopts multiple 1-D convolution layers to obtain the feature representation, where each convolution is followed by the PReLU activation to provide the non-linearity. The TCN is employed to extract long-range dependencies of encoder outputs, which utilizes stacked dilated 1-D convolutional layers with exponentially increasing dilation factor for enlarging receptive field. The output of each dilated convolution is aggregated and then passed through the gated convolution block to generate the mask with a value between 0 and 1. Different from [31], this work enforces the TCN separator to learn the mask which will be multiplied with encoder outputs to filter the noise speech features. The decoder uses the same number of convolutional layers as the encoder to transform the masked speech features into the enhanced speech waveform.

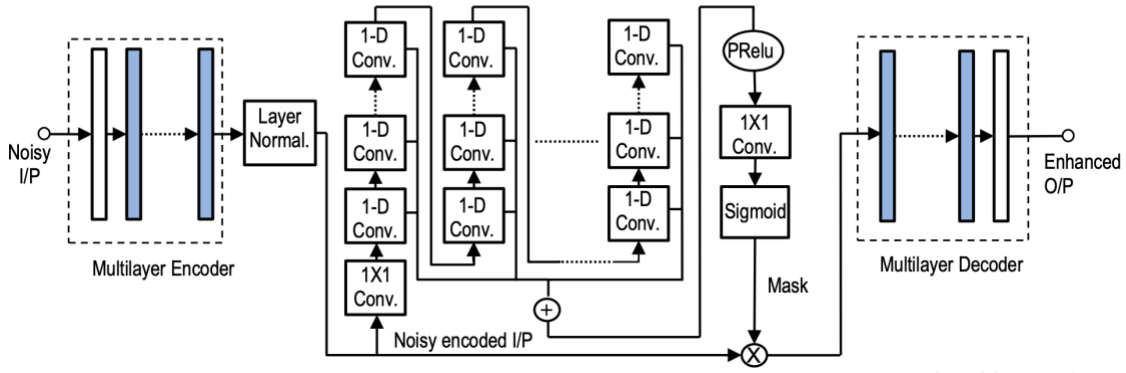


Figure 1.14: The architecture of TCN-based SE system [30]

Recently, a self-attention mechanism [32] and its variants are proposed to extract long-term dependency of sequences, which can operate in parallel compared with RNN-based models. By using the self-attention, each time step in the signal sequence will be attended by other time steps with different weight, in order to effectively capture the long-term relationship of different locations of sequence. Therefore, some researchers explored the attention-based systems for speech enhancement. For example, the authors of [33] designed a hybrid model with convolutional encoder-decoder structure and self-attention block to estimate the clean waveform from noisy speech as shown in Fig. 1.15. Like many encoder-decoder architectures, this work adopts the 1-D convolutional encoder to obtain the compressed features with the reduced sequence length. The attention block is employed to extract abundant contextual information by using the multi-head attention and feed-forward neural network. The multi-head attention divides

the whole attention procedure into multiple sub-attention steps with parallel processing, where each step proceeds the self-attention in this subspace, named as one-head attention. Then, the outputs of all one-head attentions are concatenated to form the final results of the multi-head attention, where each one-head attention generates the various features by considering different factors. For example, some heads focus more on the attentions from neighboring locations like phoneme (unit of pronunciation). In contrast, some other heads consider more the attentions from distant positions like context of the whole sequence. In [33], the masked multi-head self-attention is used to design the casual system for speech enhancement where the future information is masked in the current time step. Finally, the decoder recovers the enhanced speech by using transposed convolutions to increase the sequence length.

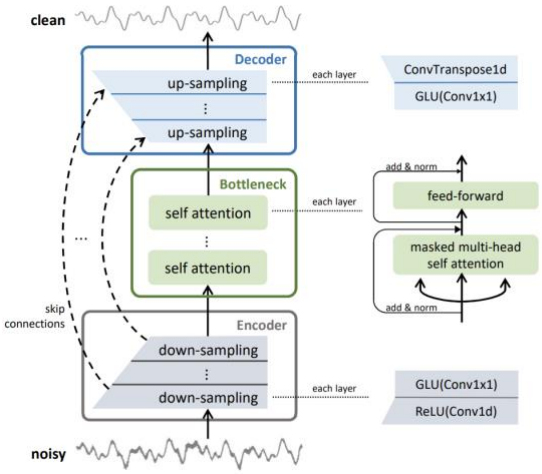


Figure 1.15: Self-attention based model for SE in the time domain [33]

1.2 Evaluation of SE models

1.2.1 Speech dataset

To evaluate the performance of SE systems, speech datasets are required to train the models, which includes training data for updating the model parameters, evaluation data for selecting the good model and testing data for evaluate the performance of the models. Some speech datasets contain both clean and noisy data with specific SNR levels for training and testing the model.

However, many datasets only include the clean utterances, which is required to mix with noises with expected SNR levels for generating the noisy speech. Some commonly used speech datasets are introduced below.

TIMIT [34]: This dataset is designed to provide the recordings for acoustic-phonetic studies and for developing and evaluating the automatic speech recognition (ASR) systems. TIMIT dataset includes the utterances of 630 speakers (438 male and 192 female speakers) with eight major dialects of American English, generating 6300 utterances in total where each ten phonetically rich utterances are completed by one speaker. The total duration of the dataset is about 5 hours, and it is recorded at 16 kHz sampling rate.

VCTK-DEMAND [35]: This dataset is selected from public Voice Bank corpus [36]. It has 28 speakers including 14 male and 14 female speakers with the same accent from England, where each speaker speaks around 400 sentences. The dataset has 11572 utterances for training and 824 utterances for testing, where all utterances are sampled at 48 kHz. To generate the noisy dataset for training, two artificially generated (speech-shaped noise and babble) and eight noise recordings from DEMAND dataset [37] are selected, including speech-shaped noise, babble, kitchen, meeting, cafeteria, restaurant, subway, car, metro and street noises. In addition, four signal-to-noise (SNR) values of 15 dB, 10 dB, 5 dB and 0 dB are applied into each noise, creating 40 different noisy conditions, where about ten different sentences are from each speaker in each condition. To create the noisy dataset for testing, two other speakers (a male and a female speaker) from England in the same corpus are selected to mix with five noises from the DEMAND dataset including living, office, bus, café and square noises. Similar to the training noisy dataset, four different SNR values are selected for noisy testing dataset, containing 17.5 dB, 12.5 dB, 7.5 dB and 2.5 dB. There are 20 different noisy conditions, where 20 different sentences are recorded from each speaker under each condition.

DNS-2020 [38]: This dataset is introduced from the deep noise suppression (DNS) challenge which is to promote the research and study real-time single-channel speech enhancement. The clean dataset is selected from the public audiobook dataset named Librivox. The dataset records about 500 hours in total from 2150 speakers, where each utterance has the duration of 10 seconds with 16 kHz sampling rate. DNS-2020 also provides noise dataset for the purpose of noisy speech generation, which is selected from Audioset and Freesound dataset.

LibriSpeech [39]: This is a large-scale audiobook selected from LibriVox. This corpus collects approximately 100 hours of speech sampled at 16 kHz, which are recorded from around 2500 speakers. Training data is categorized into three parts of 100 hours, 360 hours and 500 hours dataset while the development and testing data are divided into the 'clean' and 'other' categories, respectively, where each of development and testing dataset is about 5 hours. For speech processing task, the LibriSpeech is usually adopted for training the large-scale speech pre-trained model which is used as upstream model to evaluate the performance of various down-stream speech tasks including speech separation, speaker classification, ASR and so on.

1.2.2 Evaluation metrics

To evaluate the performance of SE models, there are two methods to evaluate the enhanced speech quality: subjective listening and objective evaluation metrics. The first method requires professional listeners to evaluate the given enhanced speech, which usually depends on the rich experience of listeners. To obtain the relative precise evaluation results, the evaluation scores from more than one listener need to be averaged, which is a little expensive and time-consuming. The second method is more convenient for researchers, which directly adopts the proposed engines to evaluate the quality of enhanced speech based on the reference clean speech. This approach considers various aspects of quality to evaluate the processed speech, including speech perceptuality, speech intelligibility, speech distortion and so on. We will introduce some commonly used evaluation metrics for SE models as follows:

Perceptual evaluation of speech quality (PESQ) [3]: The PESQ metric aims to evaluate the perceptual quality of processed speech, which is the most popular metric calculated for the enhanced and clean speech waveform. More specifically, the enhanced and clean speech are first normalized to obtain the same level of voice energy. Then, the normalized speech waveforms are aligned along the time dimension to pass through the auditory transformation processor to get the loudness spectra of each speech signal. Next, the difference between two loudness spectra is averaged along the time and frequency to predict the subjective perceptual mean opinion score. The high PESQ value, the better performance.

Short-time objective intelligibility (STOI) [40]: The STOI measurement is used to evaluate the intelligibility performance of enhanced speech, which reflects whether the speech content can

be understandable. This metric is calculated by averaging the correlation of short-time temporal envelopes between the enhanced and clean speeches. The STOI metric has consistent correlations with human intelligibility, which is usually regarded as the second commonly used evaluation metric for SE models.

Segmental signal-to-noise ratio (SSNR) [41]: The SSNR is adopted to measure the speech distortion, which is computed based on the segments of clean and enhanced speech signals. By calculating the energy and SNR of each speech segment, the averaged value of segmental SNR (dB) can be obtained as follows:

$$SegSNR = \frac{10}{N}\sum_{n=0}^{N-1} log_{10}\frac{\sum_{s=Sn}^{Sn+S-1}\|x(s)\|^2}{\sum_{s=Sn}^{Sn+S-1}\|\hat{x}(s)-x(s)\|^2} \quad (8)$$

where S and N denote the number of segments and the segment size, respectively.

Composite mean opinion scores (MOSs) [42]: The MOSs include the CSIG for evaluating the signal distortion, CBAK for distortion evaluation of background noise and COVL for overall evaluation of speech quality. Three MOSs are usually combined to form the composite objective measurements that are highly correlated with pre-set subjective ratings, which could indicate various characteristics of the distortions shown in the enhanced speech.

1.3 Essential components for training

After designing the neural networks, we need to train and optimize the models to minimize the loss between estimated output and ground truth, which is then used to update the model parameters by back-propagation method using gradient descent. To make the models converge, some essential components playing important roles during the training will be introduced below.

1.3.1 Activation function

Neural networks are regarded as a complex function to build the mapping between inputs and outputs with high-level non-linearity representation. Therefore, a non-linear activation function is used to provide non-linearity for the model to make it more powerful to learn complicated problems. Meanwhile, the activation functions must be differentiable so as to adjust the weight

and bias of the model by backpropagation. Some popular activation functions will be introduced below.

Rectified linear unit (ReLU) [9]: The ReLU is one of the most popular activation functions in neural networks, which outputs the non-zero values for positive inputs and produces the zeros for negative inputs as shown Fig. 1.16. That means the neurons can be activated only if the inputs are positive, which could bring the sparsity to networks and avoid the gradient vanishing problem caused by the depth of network. Although the gradient at zero point does not exist in mathematics, we can simply set the gradient as 0, which has no impact on the backpropagation process.

Sigmoid activation function: The sigmoid function aims to constrain the outputs between zero and one, as shown in Fig. 1.17. The sigmoid is a smooth function in terms of large inputs, whose curve tendency becomes very flat so that the gradient is almost zeros. Therefore, the gradient vanishing problem could happen at gradient value closing to zero, where the weight and bias are not updated. In addition, the sigmoid function is computationally expensive since it is of exponential form. In the SE system, the sigmoid activation function is usually used to generate a ratio mask in T-F domain masking method.

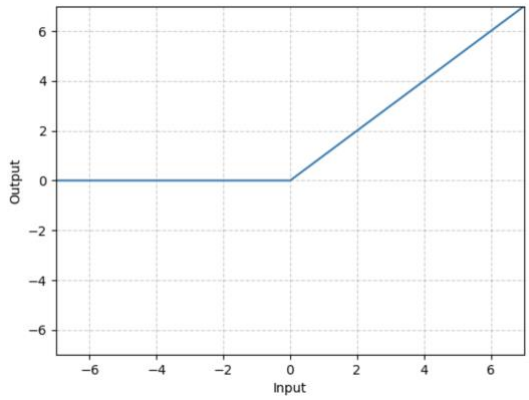


Figure 1.16: ReLU activation function

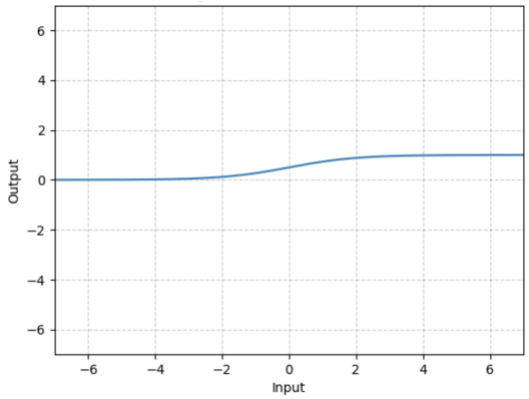


Figure 1.17: Sigmoid activation function

Gaussian error linear unit (GELU) [43]: The GELU is developed based on the ReLU activation function. Different from the ReLU, the GELU cannot output the absolute zeros if the inputs are negative as shown in Fig. 1.18. When receiving the decreasing inputs, the possibility of outputting the zeros will be enlarged. The GELU has advantages of ReLU and can avoid the vanishing gradient problem, which has been widely used in transformer-based neural networks.

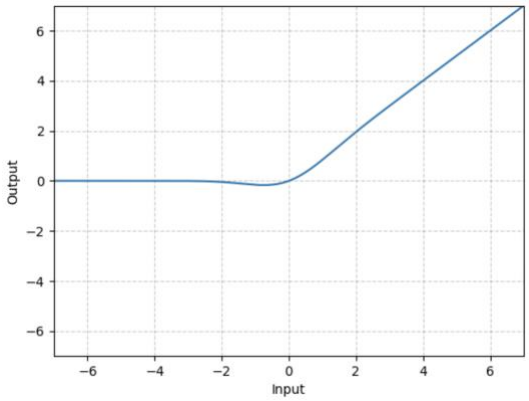


Figure 1.18: GELU activation function

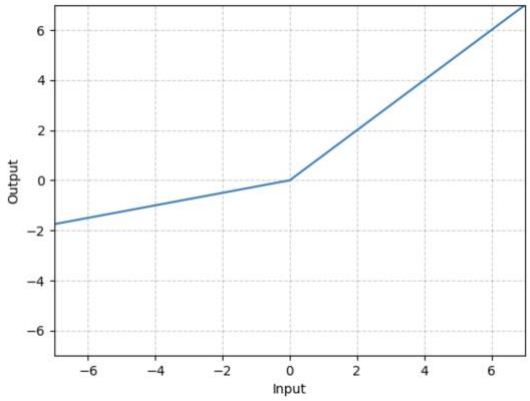


Figure 1.19: PReLU activation function

Parametric rectified linear unit (PReLU) [21]: The PReLU is an improved version of ReLU, which can solve the gradient vanishing problem caused by the negative inputs. As shown in Fig. 1.19, the PReLU multiplies the negative inputs by a non-zero value, which will generate a gradient to update the weight and bias of the model. Like ReLU, the PReLU maintains the positive inputs as is. In PReLU, the non-zero value is a trainable parameter, which is learnt to get the suitable slop to provide the gradients for weight and bias adjustment.

Softmax activation function: The softmax function takes the vector as the input and normalizes it into a probability distribution consisting of the probability of each component. After applying the softmax, each component of the vector will be constrained between 0 and 1 based on its value, and all components will add up to 1. That means the smaller components will be converted to smaller probabilities while larger ones will correspond to larger probabilities. In classification tasks, the softmax function is usually adopted as the last non-linear function of a neural network to predict the estimated classes. In addition, the attention mechanism employs the softmax to assign the weight of each component for attention score matrix.

1.3.2 Normalization

Neural network is a data-driven approach, which requires a large amount of data to update the model parameters including weight and bias. However, the dataset usually has the features of different value ranges and a small number of singular samples, which might have a bad influence on the convergence during the training. Various normalization methods are applied to solve the

model learning challenge by enforcing various features have similar feature scaling, which could make the gradient descent converge faster.

Batch normalization (BN) [18]: BN normalizes the features within the mean and variance calculated for a mini batch as demonstrated in Fig. 1.20, which has presented an effective performance on converging very deep networks in image processing. However, since BN performs on the batch dimension, it requires to work on a sufficiently large batch size to obtain accurate estimation of mean and variance, which leads to excessively large memory. In addition, it is difficult for BN to perform on the data with different lengths, like sequence modeling.

Layer normalization (LN) [44]: LN is designed to overcome the drawbacks of BN, which operates along the channel dimension instead of constraining on the batch size as presented in Fig. 1.20. More specifically, the LN is used to compute the normalization statistics from the summed input features within a hidden layer so as to remove the dependency on the batch, making it suitable for RNN and transformer layers

Group normalization (GN) [45]: GN is also adopted to normalize the features along the feature dimension as shown in Fig. 1.20. Unlike LN, the GN first divides the features into several groups and then normalizes each group individually. The GN also removes the dependency on the batch and can be employed into sequence modeling like RNN-based and transformer-based models.

Note that, the LN and GN are commonly used in SE task since the SE performs the sequence modeling, where each speech sample might have different sequence length. In addition, the RNN and transformer layers are usually used for SE systems to work with the LN or GN to make SE models converge faster.

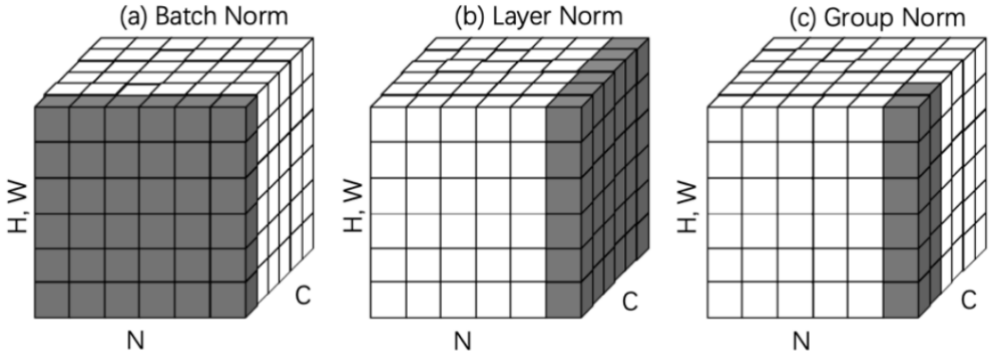


Figure 1.20: Comparisons between BN, LN and GN [46]

1.3.3 Regularization

For deep learning-based approaches, it is important for trained models to obtain good performance in unseen dataset drawn from the same distribution as the training data, which is the generalization ability. It usually happens that the performance on the training data is better or super better than that on the unseen testing data, causing the overfitting problem. The regularization technologies are employed to prevent overfitting and improve the generalization ability of the model.

Weight decay: The weight decay method adds a penalty term into the loss function, bringing an impact on shrinking the model weights during backpropagation. In practice, the weight decay factor is multiplied with the sum of weight squares, which can be incorporated into the weight update rule to produce the extra gradient. This method is used to avoid the overfitting and gradient explosion problem

Dropout [47]: The dropout is a technique in which certain neurons are randomly selected to be turned off during the training. That means the contributions of these neurons are temporally removed during forward propagation and the weight updates related to them are not executed on the backpropagation. Note that, the dropout is only applied during training stage. During the testing stage, all neurons of network always work. This technique is commonly used to avoid overfitting problem and make the model converge faster.

Residual connection [48]: Residual connection provides another path for outputs from previous layers to reach latter layers of network by skipping some intermediate layers, as shown in Fig. 1.21. The residual connection can always generate a gradient value of 1 during backpropagation, which can effectively prevent the gradient vanishing caused by obtained zero gradient. This technique is easy to implement in any neural networks to make them deeper without involving any parameters. Currently, the residual connection has been widely used in CNNs, RNNs, attention-based transformer and so on.

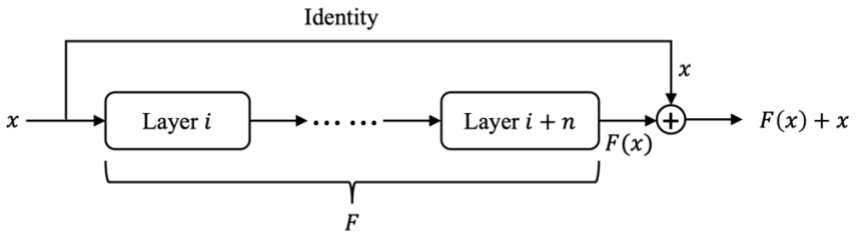


Figure 1.21: Residual connection

Note that, these regularization techniques are employed in our work as a part of the proposed neural network for improving the performance of SE systems.

1.3.4 Loss function

To train the model, we need to design a loss function or objective function to evaluate the difference between estimated outputs and ground truth. If the prediction deviates much from the ground truth, the loss function will learn to reduce the error of prediction under the help of the optimization stage. The selection of loss function depends on the specific tasks, which are broadly categorized into regression losses and classification losses. Considering the fact that SE task is a regression problem, we will introduce some loss functions commonly used in SE as follows.

Mean absolute error (MAE): MAE loss (or named as L1 loss) is calculated on the average of sum of absolute differences between predictions and ground truth, which can be formulated as:

$$MAE_{loss} = \frac{1}{N}\sum_{i=1}^{N} |\hat{y}_i - y_i| \tag{9}$$

where $\hat{y}_i$ and y_i denote the prediction of the model and ground truth, respectively, N means the number of samples.

Mean square error (MSE): MSE loss (or named as L2 loss) is measured as the average of sum of squared differences between predictions and ground truth, which can be formulated as:

$$MSE_{loss} = \frac{1}{N}\sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{10}$$

Signal distortion rate (SDR): The SDR loss is usually used for speech enhancement and separation, which is defined as:

$$loss_{SDR} = 10log_{10}\frac{||\hat{y}_i||^2}{||\hat{y}_i - y_i||^2} \tag{11}$$

where $\hat{y}_i$ and y_i denote the prediction of the model and ground truth, respectively, In the SE, the $\hat{y}_i$ and y_i present the enhanced speech and clean speech, respectively. Note that, the higher the SDR loss, the better enhanced the speech quality.

1.4 Objective and organization of thesis

The objective of this thesis is to propose novel neural networks for single-channel speech enhancement. In the first contribution, we propose two transformer neural networks with cascaded and cross-parallel structures, named as two-stage transformer and cross-parallel transformer. The former structure is adopted to successively learn the local information from each frame and global information across different frames, obtaining a better contextual representation of long-range speech sequence. To prevent the information leakage when shifting from one path to the other, the cross-parallel transformer is proposed to parallelly extract local and global information, whose outputs are then fused by a cross-attention based transformer to generate efficient contextual features. The outputs from transformer blocks are passed through a masking module to create a mask, which is used to multiply with encoder outputs to remove the redundant noise features. Our proposed models present an impressive performance of SE compared with existing systems while involving the fewest trainable parameters.

Although many competitive performances are realized by networks involving attention mechanism or transformer as a part of them, it is not thoroughly studied as to whether the attention is indispensable for speech enhancement, especially when considering the trade-off between computational efficiency and denoising performance. In the second contribution, we propose an attention free architecture named SE-Mixer by using MLP as backbone. The proposed SE-Mixer uses mixer blocks to perform temporal and frequency mixing, where each block includes a temporal Conv-MLP and a frequency MLP to iteratively extract the temporal information from different time steps and the frequency information within each time step. We have also shown that the architecture without employing attention algorithm is likely to reach approximately the same effectiveness as compared with attention-assisted models but with significantly lower computational complexity.

The rest of this thesis is organized as follows:

Chapter 2: This chapter first introduces the architecture of transformer and some transformer-based models for speech enhancement. It then describes the proposed two-stage transformer and cross-parallel transformer for speech enhancement in the time domain. Finally, the experimental results of proposed transformer-based models are presented, including the

results of comparison with existing methods and ablation study of proposed models with different configurations.

Chapter 3: This chapter first describes the attention-free MLP-Mixer architecture, based on which a new Mixer framework is presented for speech enhancement. The proposed Mixer model adopts the temporal MLP and the frequency MLP to perform the temporal mixing and frequency mixing, generating a contextual information of speech signal. Experimental results of the proposed approach are provided, demonstrating the attention-free models could achieve competitive performance of speech enhancement compared to attention- or transformer-based models yet with reduced computational complexity.

Chapter 4: This chapter concludes the thesis and suggests some directions for future work.

Chapter 2
Transformer-based neural networks for speech enhancement

In this chapter, we propose a two-stage transformer neural network and a cross-parallel transformer neural network for speech enhancement, abbreviated as TSTNN and CPTNN, respectively. This chapter is organized as follows. We first overview the vanilla transformer structure in Section 2.1. Based on this, the recent works using transformer-based networks for speech enhancement and separation are introduced in Section 2.2. In Section 2.3, we describe the proposed TSTNN and CPTNN for signal-channel speech enhancement in the time domain. And Section 2.4 provides the evaluation performance of the proposed TSTNN and CPTNN.

2.1 Introduction

Recently, transformer neural network is proposed to extract the long-term dependency of sequences, which has achieved an impressive performance in NLP and CV tasks. Many researchers have studied the transformer-based architectures in speech processing tasks including speech enhancement, speech separation, speech recognition and speech synthesis. In this section, we first introduce the overall structure of vanilla transformer and its components, and then review some speech enhancement and separation works based on transformer.

2.1.1 Overview of vanilla transformer

The vanilla transformer neural network is originally proposed in [32] to extract long-range dependencies using self-attention mechanism in sequence modeling. Meanwhile, the transformer can operate well in parallel compared with sequential RNN-based models. As shown in Fig. 2.1, the vanilla transformer consists of an encoder and a decoder, where the encoder first transforms inputs into feature embeddings and then the decoder decodes these high-level feature

embeddings into outputs. More specifically, the transformer model is auto-regressive at each step, where the input is a sequence of symbol representations. And, the output is probability distribution representing one element of symbols at a time, which will be used as additional input of decoder to generate the next-step output.

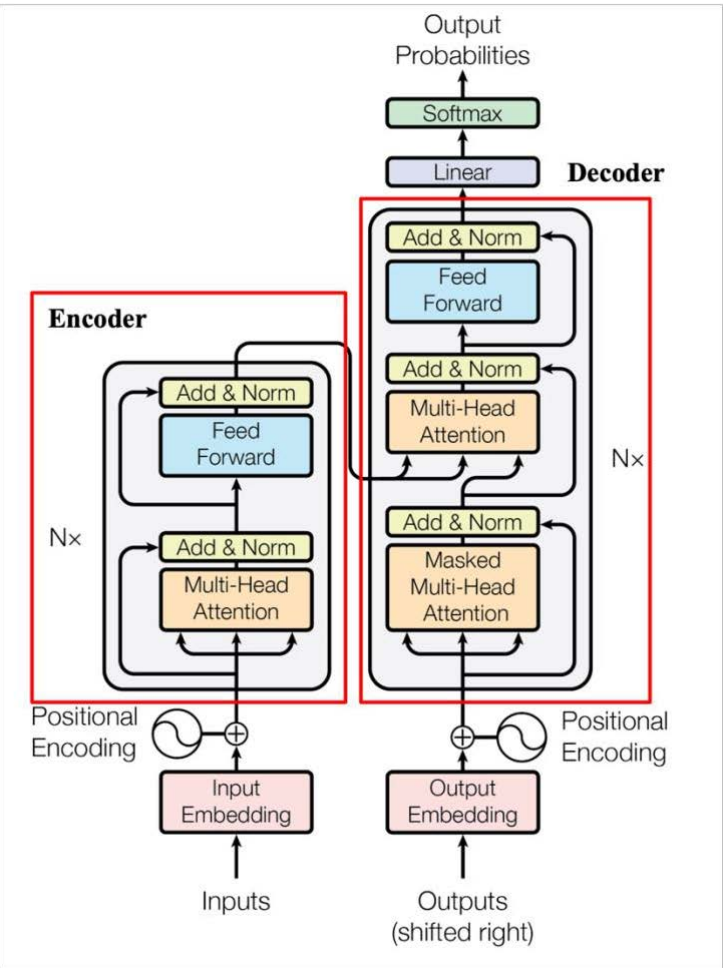


Figure 2.1: The overall architecture of transformer [32]

Transformer encoder and decoder

The transformer encoder is comprised of a stack of N identical layers, where each layer involves multi-head attention (MHA) mechanism and position-wise fully connected feed-forward network (FFN). Both MHA and FFN are followed by residual connection and layer normalization. In the beginning of encoder, the positional encoding (PE) is adopted to input embeddings to represent the positional information. Similar to encoder, the transformer decoder is composed of a stack of

N identical layers, each involving three sub-layers including MHA, masked MHA and FFN. The masked MHA is used to enforce that the output of current time step is only decided by itself and its previous steps by masking the subsequent time steps, which can preserve the auto-regressive property of transformer. The MHA of decoder performs the attention mechanism between the outputs of transformer encoder and masked MHA layer to help decoder generate outputs by making use of information from encoder. Like the encoder, the residual connections and layer normalization are employed at each sub-layer, and the positional encoding is added into output embeddings to learn the positional information.

Multi-head attention

The attention algorithm is used to focus on what we want just like human, which can be described as mapping three vectors of query, key and value to an output. As shown in Fig. 2.2, the input features are linearly transformed into three different feature representations, including queries and keys with dimension d_k, and values with dimension d_v. The dot products of the queries and keys are divided by the scaled factor $\sqrt{d_k}$, and then processed by the softmax function to obtain the weight matrices of values. Finally, the attention is performed by multiplying the weight matrices with values. The procedures can be written as:

$$Q = XW^Q, K = XW^K, V = XW^V \tag{12}$$

$$Attention\ (Q, K, V) = softmax(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}})V \tag{13}$$

where X presents the inputs with feature dimension of d_{model}, $Q, K,$ and V denote the queries, keys and values, and W^Q, W^K, W^V are linear transformation matrices for queries, keys and values, respectively.

Instead of using a single attention function with keys, values and queries. It is useful to linearly project the queries, keys and values by h times using different and learnable linear transformations to produce h projected versions of keys, values and queries as shown in Fig. 2.3, where each one performs scaled dot product attention in parallel, regarded as one head attention. In the following, the generated multiple head attentions are concatenated and projected again with one linear layer to obtain the outputs of multi-head attention. Different from one head attention, the multi-head attention can make model to learn different representations of subspace

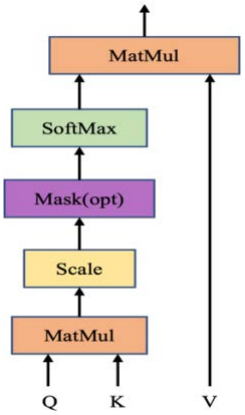


Figure 2.2: Scaled dot-product attention

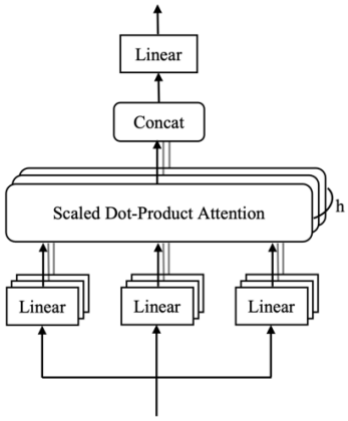


Figure 2.3: Multi-head attention

at different positions. The procedures can be formulated as follows:

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \quad (14)$$

$$head_i = Attention\ (XW_i^Q, XW_i^K, XW_i^V) \quad (15)$$

where W_i^Q, W_i^K, W_i^V present the ith linear transformation matrix to obtain the queries, keys and values, respectively, and $i = 1, 2, 3 \dots, h$, and h is the number of heads, W^O is the linear transformation matrix to output the results of multi-head attention.

The MHAs are used in both encoder and decoder layers to perform different roles. More specifically, the encoder layer uses one MHA to allow each position of sequence to be attended by other positions. However, the decoder layer employs two different versions of MHA, where the first one only allows the communications among previous positions of each time step by masking its future position information and the second one receives the encoder outputs as keys and values, and outputs of masked MHA as queries for building the communication between encoder and decoder.

Position-wise fully connected feed-forward network

The position-wise fully connected feed-forward network (FFN) is applied to each position of sequence separately and identically, which includes two fully-connected layers with a ReLU non-linearity activation in between. The procedure can be formulated as follows:

$$FFN(x) = ReLU\ (xW_1 + b_1)W_2 + b_2 \quad (16)$$

$$ReLU\ (z) = \max(0, z) \quad (17)$$

where x is the input of FFN, $W_1 \in \mathbb{R}^{d \times (4d)}$ and $W_2 \in \mathbb{R}^{(4d) \times d}$ are linear transformation matrices of FC layers, $b_1 \in \mathbb{R}^{4d}$ and $b_2 \in \mathbb{R}^{d}$ are bias for FC layers, $ReLU(\cdot)$ is ReLU non-linearity operation.

Positional Encoding

Positional encoding (PE) is added to the inputs of transformer to represent the positional information, since the multi-head attention operation performs on the whole sequence without considering the order of the sequence. The PE has the same dimension as the input embeddings so that they can be summed. In vanilla transformer, the sine and cosine functions of different frequencies are used as PE, shown as follows:

$$PE(pos, 2i) = \sin\left(pos/10000^{2i/d_{model}}\right) \tag{18}$$

$$PE(pos, 2i+1) = cos(pos/10000^{2i/d_{model}}) \tag{19}$$

where pos denotes the position and i is the dimension.

2.1.2 Existing transformer-based speech enhancement and separation

Inspired by the effectiveness of transformer neural network in extracting long-range dependencies using self-attention mechanism, some researchers proposed transformer based neural network for promoting the performance of speech enhancement and separation. Authors of [49] proposed a transformer with gaussian-weighted self-attention for speech enhancement, where the attention block of vanilla transformer is modified by using a gaussian weighting matrix to multiply with the attention scores. As indicated in Fig. 2.4, the gaussian-weighted self-attention block has the similar structure as original multi-head attention block except that the attention matrix is scaled by the gaussian weighting matrix. The attention matrix is computed from the matrix multiplication between key and query. Before passing through the softmax operation, the attention matrix is multiplied with a gaussian weight matrix with an element-wise manner. The procedures can be formulated as follows:

$$AttentionScore\ (Q, K) = softmax\left(\frac{QK^{\mathrm{T}}}{\sqrt{d_k}} \cdot G\right) \tag{20}$$

$$G = \begin{bmatrix} g_{1,1} & \cdots & g_{1,T} \\ \vdots & \ddots & \vdots \\ g_{T,1} & \cdots & g_{T,T} \end{bmatrix} \tag{21}$$

where G is the gaussian weight matrix with its element being given by $g_{i,j} = e^{\frac{-|i-j|^2}{\sigma^2}}$, where i denotes the target frame index, j means the context frame index, and σ is the trainable parameters to decide the weight variance. Note that, the diagonal elements in G is to scale the target frames themselves, which are set to be 1. However, other elements are inversely proportional to the distance between the target frames and context frames, which provides less attention to the more distant context frames and more attention to the closer ones.

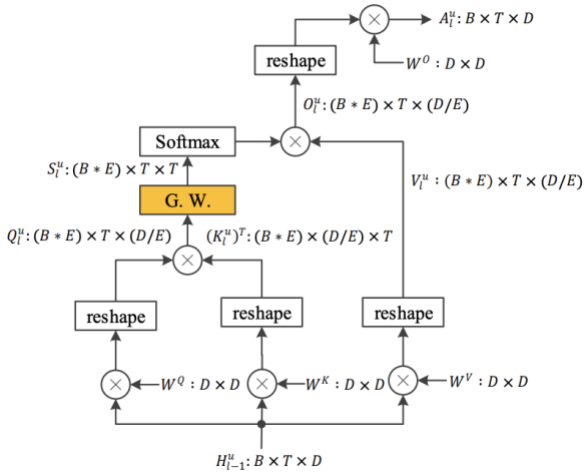


Figure 2.4: Gaussian-weighted self-attention [49]

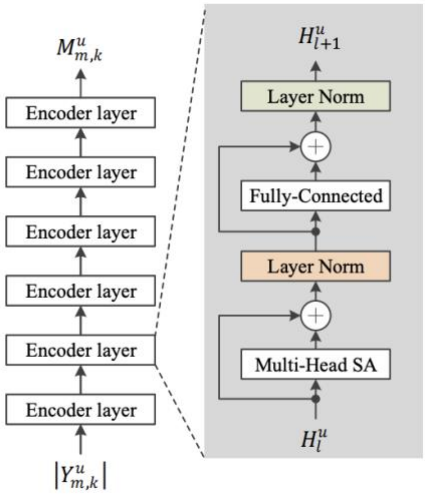


Figure 2.5: The structure of T-GSA [49]

Based on the modified multi-head self-attention block, the authors in [49] adopt multiple identical transformer encoder layers to estimate the clean spectrogram magnitude from noisy spectrogram magnitude as shown in Fig. 2.5. The enhanced speech is obtained by the estimated magnitude as well as the phase of noisy speech via inverse short-time Fourier transform.

Authors of [50] introduce the convolution-augmented transformer (conformer) to enhance the speech signal, where the convolution is inserted between MHA and FFN to learn the local contextual information. As shown in Fig. 2.6, the conformer block includes MHSA, macaron-like FFN module, convolution module and layer normalization. Different from the vanilla transformer, the FFN modules have two FFNs to sandwich the MHSA and convolution module, where each FFN contains two FC layers and swish activation in between. The convolution

module is comprised of two point-wise convolutions and a 1-D depth-wise convolution in between, where the GLU and swish activation are applied after the first convolution and before the last convolution, respectively. In this work, the conformer block is inserted between encoder and decoder in Fig. 2.7, where the encoder encodes the feature representation of speech waveform and the conformer extracts the global contextual information by MHSA and local contextual information by convolution module, and decoder recovers the contextual information back enhanced speech waveform.

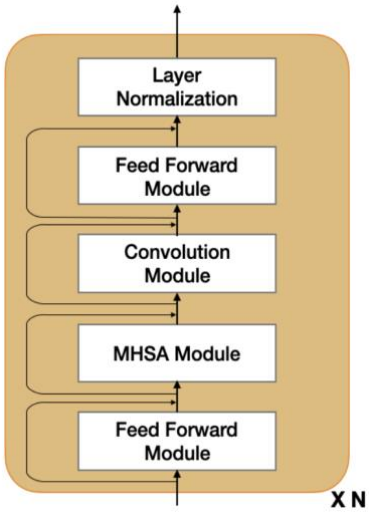


Figure 2.6: Conformer structure [50]

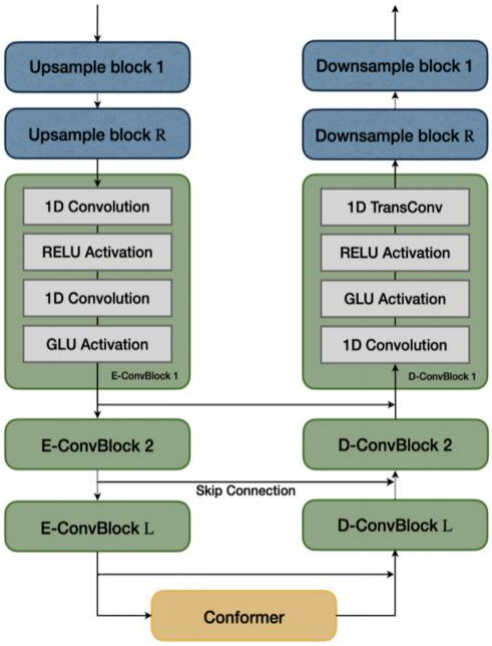


Figure 2.7: The structure of SE-Conformer model [50]

In the waveform domain, the input speech sequences are usually super long so that the transformer cannot process it due to the heavy computational memory. Inspired by the dual-path framework proposed by [51], the authors in [52] proposed a separation transformer (SepFormer) for speech separation to learn the short-term and long-term dependencies of speech sequences. The SepFormer consists of an encoder, a masking network and a decoder. The encoder takes the speech waveform as input to learn the STFT-like representation which passes through the masking network to model the contextual information. As indicated in Fig 2.8, in the masking network, the inputs are first normalized by layer normalization and processed by a linear layer to get features. Then, the chunking stage splits the 1-D features into over-lapped chunks, which are then packed together to form the 2-D features. Next, the sepformer block adopts the intra

transformer and inter transformer to extract the short-term and long-term information, respectively. More specifically, the intra transformer performs within each chunk to learn the short-term dependencies. The inter transformer is applied into permuted features to learn the relationship cross chunks, resulting in the long-term dependency modeling. After multiple sepformer blocks, the features are procced by a PReLU activation, a linear layer and an overlap-add stage to get 1-D features, which is followed by two FC layers and a ReLU activation function to obtain the mask for the separation of each speaker. Finally, the decoder recovers each separated speech feature back to speech waveform by using a transposed convolution. By using the dual-path pipeline including transformer for short- and long-term modeling, the SepFormer achieves an impressive separation performance with less computational complexity.

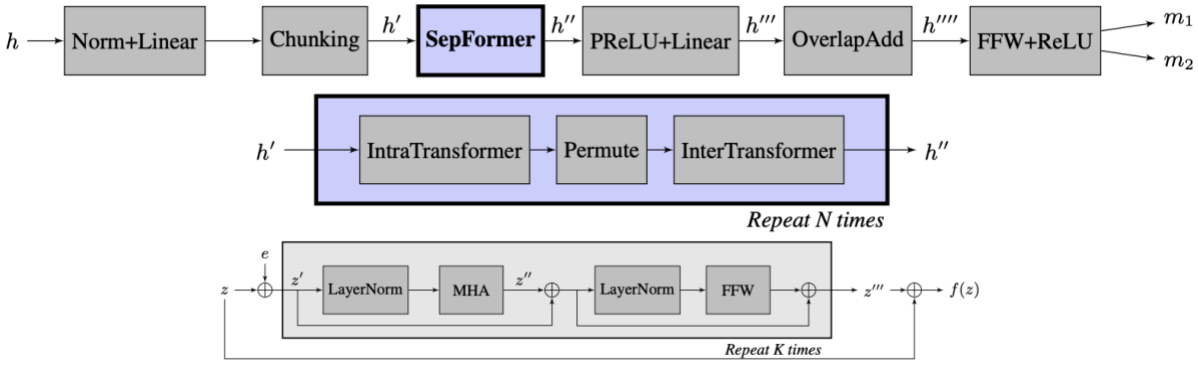


Figure 2.8: The architecture of SepFormer for speech separation [52]

2.2 Proposed transformer cores

In this section, we introduce our two proposed transformer blocks, namely, two-stage transformer block (TSTB) and cross-parallel transformer block (CPTB). In order to extract the contextual information of speech features, a dual-path structure is employed in the two transformer blocks. The TSTB uses the cascaded local and global extraction to successively capture the local and global information. In contrast, the CPTB adopts the parallel local and global extraction to simultaneously learn the local and global information, which will be fused by cross-attention based fusion block for adaptive features.

2.2.1 Two-stage transformer

We first present an improved version of the general transformer structure, based on which we propose a two-stage transformer block (TSTB) for extracting local and global contextual information of speech feature by incorporating dual-path structure.

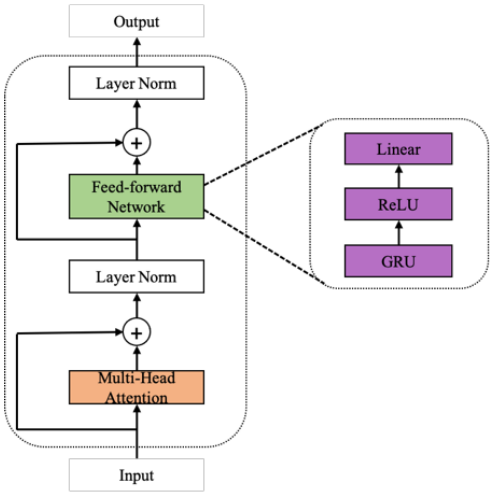


Figure 2.9: Improved transformer encoder

The general transformer structure consists of encoder and decoder networks. In our proposed TSTB, we only use the encoder part since the input mixtures and output enhanced sequences have the same length in speech denoising. The original transformer encoder is comprised of three important modules including PE module, MHA module and position-wise FFN. Different from that, we remove the PE part since it is not suitable for acoustic sequence. Inspired by the effectiveness of RNNs in tracking order information, the first fully connected layer of FFN is replaced with a GRU layer to learn the positional information, as shown in Fig. 2.9.

In MHA block, first, the input (X) is mapped with different, learnable linear transformation h times to get queries (Q), keys (K) and values (V) representation, respectively, as described in Eq. (22). Then, the dot product of the query with all keys is computed, followed by division of a constant. After applying the softmax function, the weights on the values are obtained. The attention of each head is the dot product of the weights and values as shown in Eq. (23). The attentions of all heads are concatenated and linearly projected again to obtain the final output in Eq. (24), which is followed by layer normalization and residual connection of input X as given by Eq. (25).

$$Q_i = XW_i^Q\ ,\ K_i = XW_i^K\ , V_i = XW_i^V \tag{22}$$

$$head_i = Attention(Q_i, K_i, V_i) = softmax(\frac{Q_i {K_i}^T}{\sqrt{d}})V_i \tag{23}$$

$$MultiHead(Q, K, V) = Concat(head_1, \dots, head_h)W^O \tag{24}$$

$$Mid = LayerNorm(X + Multihead) \tag{25}$$

where $X \in \mathbb{R}^{l \times d}$ is the input with sequences of length l and dimension d, $i = 1, 2, \dots h$ and $Q_i, K_i, V_i \in \mathbb{R}^{l \times d/h}$ are the mapped queries, keys and values respectively, $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d \times d/h}$ denote the ith linear transformation matrix for queries, keys and values, respectively. $W^O \in \mathbb{R}^{d \times d}$ is the linear transformation matrix and h is the number of parallel attention layers which is set as 4 in our proposed transformer.

Then, the output from MHA block is processed by GRU-augmented FFN to obtain the final output of improved transformer encoder, where residual connections and layer normalization are added as well. The procedures are defined as follows:

$$FFN(Mid) = ReLU(GRU(Mid))W_1 + b_1 \tag{26}$$

$$Output = LayerNorm(Mid + FFN) \tag{27}$$

where $FFN(\cdot)$ denotes the output of the position-wise feed-forward network, and $W_1 \in \mathbb{R}^{d_{ff} \times d}$, $b_1 \in \mathbb{R}^d$, and $d_{ff} = 4 \times d$.

Although transformer achieves an impressive performance in modeling long-range dependency, it yields huge computation due to massive matric multiplication operations when directly performed on long speech sequences. To solve the problem of long speech sequence modeling, authors of [51] proposed a dual-path structure to improve the performance of RNNs on modeling long sequence for speech separation. More specifically, the input speech sequences are first processed within a small frame window to learn local relationship. Then, the samples from different frames are aggregated to model global information. Both of the two steps are processed by RNN layers with normalization operations. Inspired by this, we propose a two-stage transformer block (TSTB) for speech enhancement as shown in Fig. 2.10. It has a local transformer and global transformer to successively extract local and global contextual information, respectively.

In the proposed TSTB, the input features X have a shape of $[B, C, N, F]$. The local transformer is first applied to individual chunks to parallelly extract local information, which performs on the last dimension F of input features. Then the global transformer is employed for

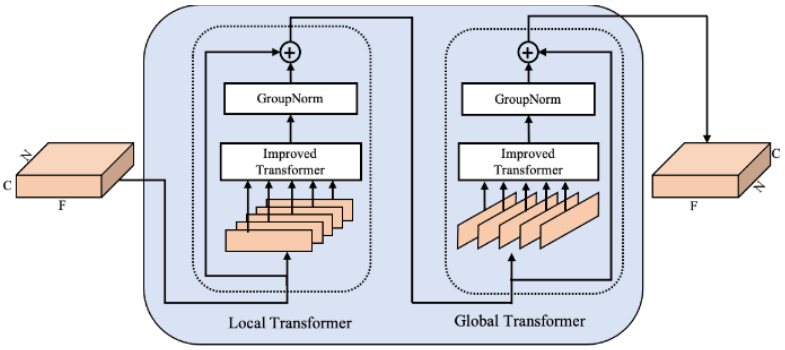


Figure 2.10: Proposed two-stage transformer block (TSTB)

fusing the information of output from local transformer to learn the global dependency of different chunks, which implements on the dimension N of features. Besides, the group normalization and residual connection are utilized into both local and global transformer to avoid overfitting and gradient vanishing.

$$Out_{local} = LocalTransformer(Reshape(X)[:,:,f]) + Reshape(X) \quad (28)$$

$$Out_{global} = GlobalTransformer(Reshape(Out_{local})[:,l,:]) + Reshape(Out_{local}) \quad (29)$$

where $l = 1, 2, \cdots, N$, denotes the index of chunks, and $f = 1, 2, \cdots, F$, means the number of samples in each chunk. $Reshape(\cdot)$ denotes the changes of feature layout, equivalent to the operation of dimension permutation.

2.2.2 Cross-parallel transformer

To tackle the potential drawback that the information delay and leakage while shifting from one path to the other one caused by the cascaded two-stage structure, we propose the parallel structure to simultaneously extract local and global information, generating the outputs of two branches will be dynamically fused by cross attention mechanism. By incorporating this new cross-parallel structure and transformer, we propose a cross-parallel transformer block (CPTB) to adaptively extract local and global contextual information without impact on each other.

To construct our proposed CPTB, we first propose two improved transformers with self-attention and cross-attention mechanism which are shown in Fig. 2.11. The improved self-attention transformer is different from the one in TSTB. First, we propose a modified FFN to learn the positional information, which includes a bidirectional GRU layer as well as a FC layer

with GELU nonlinearity in between. The ReLU activation function is commonly used in neural network to alleviate the overfitting problem. However, the ReLU may lead to the gradient vanishing in the case of having negative inputs. Different from ReLU, the GELU activation function possesses the benefits of ReLU while avoiding gradient vanishing. Second, we apply the pre-norm strategy [53] to train a good transformer by adopting layer normalization before the

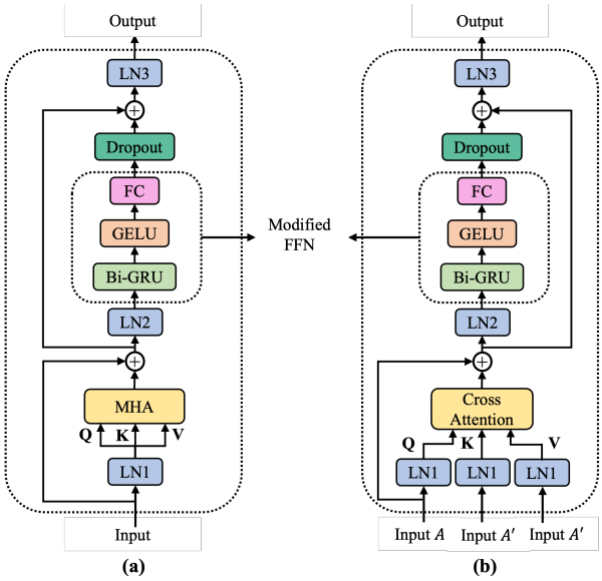


Figure 2.11: Overview of two proposed improved transformers. (a) transformer with self-attention mechanism; (b) transformer with cross-attention mechanism (Note that 'LN' means the layer normalization)

MHA module and modified FFN. Authors of [54] have verified that the pre-norm based transformer has a similar and even better performance than post-norm based one while avoiding the complex warm-up preprocessing. Finally, an additional layer normalization is employed at the end of transformer to avoid the by-passing problem [54].

The cross-attention based transformer applies the cross attention whose structure is the same as the MHA of the improved self-attention based transformer, except that the queries (Q), keys (K) and values (V) are obtained from different input features as shown in Fig. 2.11 (b). More specifically, Q comes from input feature A, however K and V are obtained from input feature A'. The cross-attention mechanism guides the transformer to allow the communication between A and A'during training, which makes an efficient fusion of different features.

Based on the two improved transformers proposed above, we here propose a cross-parallel transformer block (CPTB) with a local transformer, a cross transformer and a global transformer

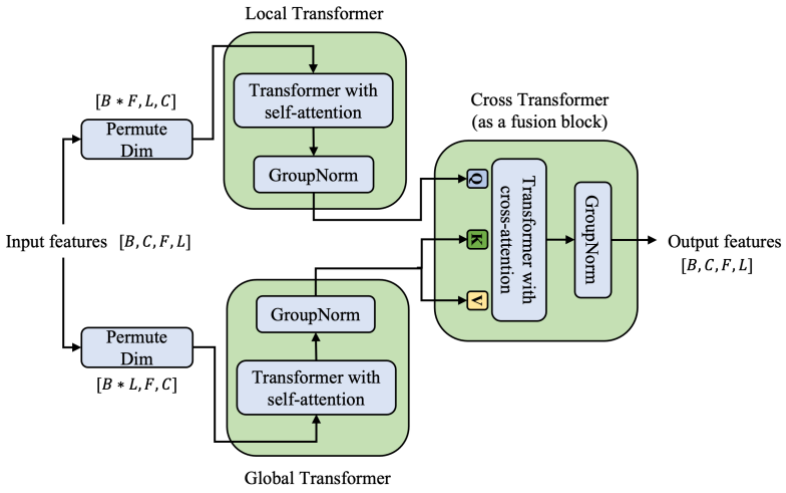


Figure 2.12: Proposed cross-parallel transformer block (CPTB). (B denotes the batch size; C, F and L are the channel, the number of frames and the frame size, respectively)

as shown in Fig. 2.12. The local and global transformers both based on the self-attention are utilized for parallelly extracting the local and global information, respectively. Their outputs are fused by a cross transformer based on the cross-attention. More specifically, the input tensor I has a size of $[B, C, F, L]$, and the local transformer is employed on each independent frame to extract the feature information, which implements along with the dimension L of the input tensor I as shown in Eq. (30). Simultaneously, the global transformer aggregates the feature information from different frames to learn the global dependency of speech sequences by performing on the dimension F of the tensor I as described in Eq. (31). Then, the cross transformer, as a fusion block, is used to fuse the output features from local and global transformers using cross-attention mechanism as given by Eq. (32). Finally, the group normalization [45] is adopted following each transformer for regularization.

$$Out_{local} = LocalTransformer(Reshape(I)[:,:,l]) \tag{30}$$

$$Out_{global} = GlobalTransformer(Reshape(I)[:,f,:]) \tag{31}$$

$$Out_{cross} = CrossTransformer(Out_{local}, Out_{global}) \tag{32}$$

where $l = 1, 2, \cdots, L$, means the number of samples in each chunk, and $f = 1, 2, \cdots, F$, denotes the index of chunks. $Reshape(\cdot)$ means the shape changes of feature dimension, which is equal to the operation of dimension permutation as shown in Fig 2.12.

2.3 Proposed transformer neural networks

In this section, we propose two transform-based models for speech enhancement, namely two-stage transformer neural network (TSTNN) and cross-parallel transformer neural network (CPTNN). The TSTNN adopts the proposed two-stage transformer blocks to successively extract local and global contextual information of speech sequences. On the other hand, the CPTNN employs the cross-parallel transformer blocks to parallelly capture local and global information, which are then fused by the inner cross-attention transformer to generate the efficient contextual representation.

2.3.1 Proposed architecture

Our proposed TSTNN and CPTNN share similar components including the encoder, masking module and decoder, except for the intermediate transformer blocks as shown in Fig. 2.13. For example, the TSTNN adopts transformer blocks with cascaded structure, each having local and global transformers. In contrast, the CPTNN employs the transformer blocks with cross-parallel structure, where each block comprises local and global transformers in parallel and then fuses the two transformer outputs by another cross transformer. In both TSTNN and CPTNN, the noisy speech waveform is first pre-processed by to-chunk method in time domain. Then, the inputs are processed by convolution-based encoder to extract low-level features, generating the compressed features with smaller sequence length which will pass through the transformer blocks. TSTNN adopts the two-stage transformer blocks to successively extract local and global information of long-range speech sequences, yielding the feature representations considering the contextual information. Different from TSTNN, the CPTNN employs two parallel branches to capture local and global information, which are then adaptively fused by cross-attention based transformer for better contextual features. Next, the masking module receives the outputs from transformer blocks to generate a mask which will be multiplied with encoder outputs to obtain the masked encoder features, thereby suppressing most of noise features. Finally, the decoder is applied to reconstruct the masked encoder features into enhanced speech features with a similar form of encoder inputs, which are post-processed by overlap-add method in time domain for eventually producing the enhanced speech waveform.

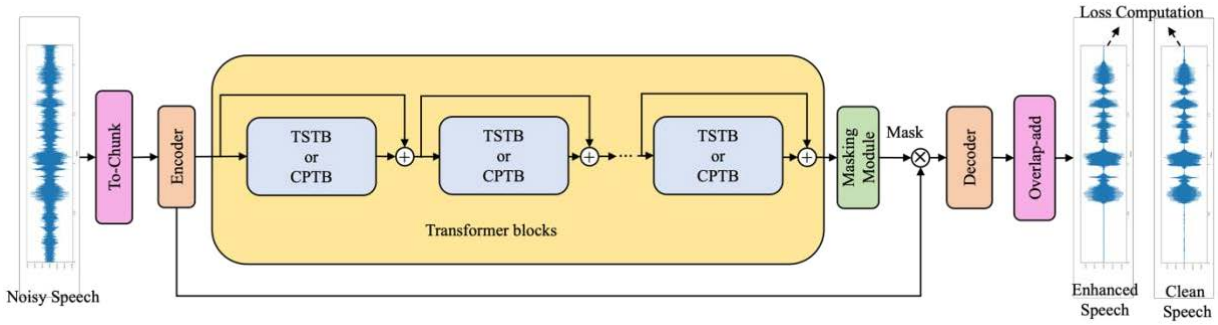


Figure 2.13: Overview of proposed TSTNN and CPTNN
(For transformer blocks, the TSTNN adopts the TSTBs, and the CPTNN employs the CPTBs)

Our TSTNN and CPTNN are proposed for time-domain speech enhancement where the input is the speech waveform which is obtained by resampling the speech audio at a certain sampling rate. For example, we can obtain 32000 samples when resampling a 2-second continuous speech signal at 16kHZ, producing the whole utterance. However, it is in general difficult to process very long speech samples for the deep learning-based models even though we can randomly select smaller speech segments within a long speech utterance. In our proposed model, we adopt a to-chunk preprocessing to transform the long speech samples into shorter sequences. More specifically, a row input mixture $X \in \mathbb{R}^{1 \times M}$ is splitted into F overlapped chunks of length L with a shift size S. These chunks are then stacked to create a 3-dimensional tensor $T \in \mathbb{R}^{1 \times F \times L}$ as shown in Fig. 2.14. Here F is defined as:

$$F = \lceil (M - L)/(L - S) + 1 \rceil \tag{33}$$

where $\lceil \cdot \rceil$ rounds the number involved up to the nearest integer. Note that for the last chunk whose size is smaller than L, zero-padding is applied to match the equal chunk size. This newly constructed input is processed by speech enhancement model, outputting the similar 3-dimensional tensor. The overlap-add stage is used as the inverse operation of to-chunk preprocessing to merge chunks for reconstructing the 3-D tensor back 1-D enhanced speech waveform.

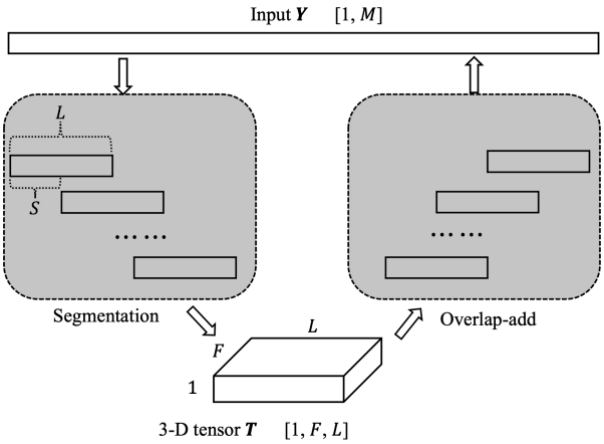


Figure 2.14: Illustration of to-chunk preprocessing

2.3.2 Encoder

The encoder is proposed to extract compressed features from inputs, which is comprised of multiple layers including one input layer, D down-sampling layers and one output layer as shown in Fig. 2.15. First, the number of channels of inputs is increased by input layer from C_{in} to 64 via using 2-D convolutions with 64 kernels of size (1, 1). After that, each down-sampling layer employs a dense block using three dilated convolution layers incorporated and the 2-D convolution with kernels of size (1, 3) at a stride of (1, 2) to halve the dimension of the frame size. Finally, the output layer halves the dimension of channel via convolution with 32 kernels of size (1, 1), thus bringing down the computational complexity for the subsequent transformer networks. In addition, the layer normalization and PReLU nonlinearity [21] are used after all convolutional layers.

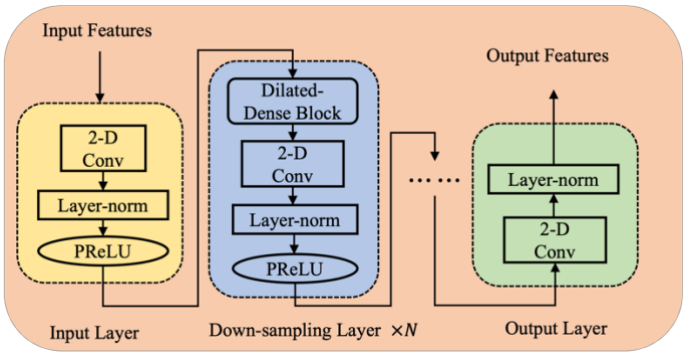


Figure 2.15: Illustration of encoder layer

Dilated-dense block is first applied in speech enhancement in [26], which is developed based on the densely connected convolutional network (DCCN) [55]. In the DCCN, powerful feature propagation is realized by adding all previous outputs to each layer, and gradient vanishing problem of deep neural networks could be alleviated in the training as well. Different from [26], all conventional convolutions in dense block are replaced by dilated depth-wise separable convolutions [56] to increase the receptive field of features as well as reduce model parameters. An illustrative structure of depth-wise separable dilated-dense block as shown in Fig. 2.16. Each dense block contains N layers of 2-D depth-wise separable convolutions with exponentially increasing dilation rates (1, 2, 4, …2^{N-1}). Each convolutional layer is followed by the layer normalization and PReLU nonlinearity.

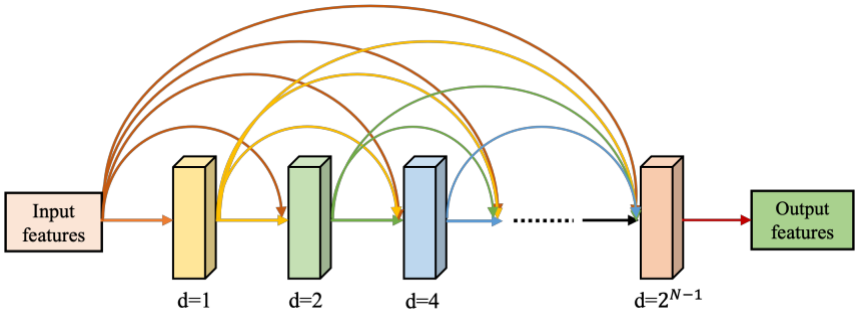


Figure 2.16: Illustration of dilated-dense block

2.3.3 Transformer blocks

Based on our proposed TSTB and CPTB, we construct intermediate transformer blocks to capture the contextual information of speech features as demonstrated in Fig. 2.17. In both TSTNN and CPTNN, the transformer blocks are inserted after encoder to extract the long-range dependency of speech sequences. More specifically, the TSTNN uses N_{TSTB} stacked TSTBs to iteratively extract local and global information for better contextual representation. The CPTNN implements N_{CPTB} stacked CPTBs, each followed by the residual connection from its input for avoiding the gradient vanishing problem. Different from the TSTB, the CPTB is adopted to parallelly extract local and global features which are then dynamically fused by inner cross-attention mechanism, leading to a global contextual feature.

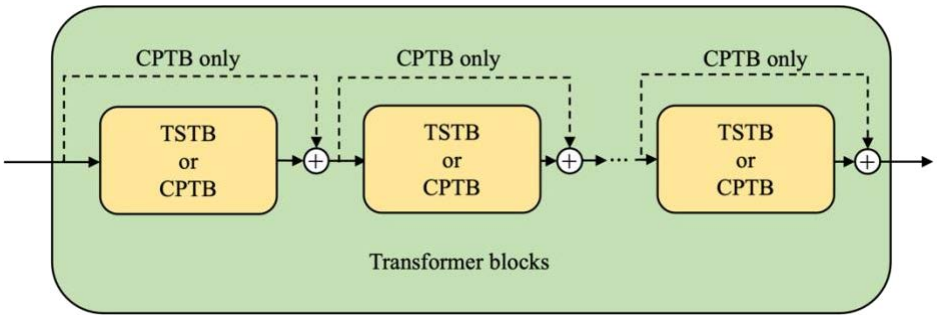


Figure 2.17: Illustration of transformer blocks used in TSTNN and CPTNN

2.3.4 Masking module

The masking module utilizes the feature output from transformer blocks to generate a mask to enhance speech, as shown in Fig. 2.18. The channel dimension of the output from transformer blocks is first doubled by a convolution followed by PReLU nonlinearity to match the dimension of the encoder output. Then it undergoes a gated convolution [28] involved a 2-D convolution and sigmoid nonlinearity operation, with its outputs passing through a 2-D convolution and ReLU nonlinearity for creating a mask. The generated mask will be multiplied with the encoder output in element-wise manner for giving final masked encoder features.

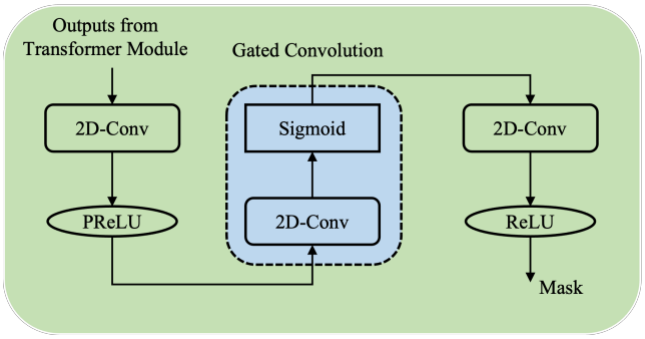


Figure 2.18: Illustration of masking module

2.3.5 Decoder

The decoder is adopted to reconstruct features from masking module into the denoised speech features, which is comprised of *D* up-sampling layers plus one output layer as shown in Fig. 2.19. Each up-sampling layer employs a dilated-dense block and a sub-pixel convolution to double the

dimension of the frame size, which is regarded as the inverse procedure of the encoder. The sub-pixel convolution similarly performs as the transposed convolution to enlarge the dimension size. Compared with the transposed convolution, it operates on the signal itself without adding zero entries and increases the size of feature by an up-sampling rate, which can resolve the checkerboard artifacts problem [57]. Finally, the output layer resumes the dimension of the enhanced speech features back to C_{in} by the 2-D convolution with a kernel size of (1, 1), followed by overlap-add method for generating the enhanced speech waveform.

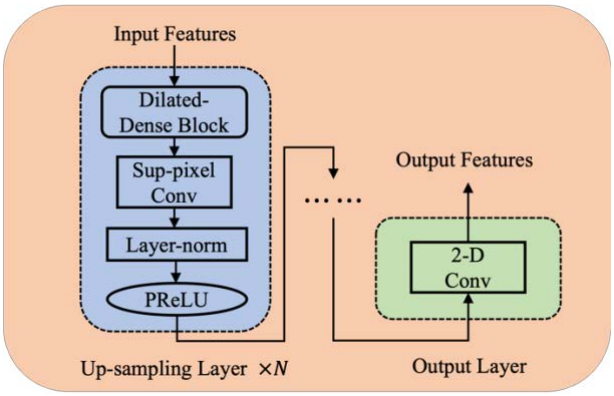


Figure 2.19: Illustration of decoder layer

2.3.6 Loss function

To train our proposed models, we adopt the loss function combining the time domain loss and time-frequency domain loss. The time-frequency domain loss enforces the neural network model to capture frequency information, making speech have higher speech intelligibility and perceptual quality [58], which is formulated as:

$$loss_F = \frac{1}{TF}\sum_{t=0}^{T-1}\sum_{f=0}^{F-1}[(|S_r(t,f)| + |S_i(t,f)|) - (|\hat{S}_r(t,f)| + |\hat{S}_i(t,f)|)] \quad (34)$$

where S and $\hat{S}$ denote the complex spectrogram representations of clean waveform and speech waveform, r and i are the real and imaginary part of the complex variable, and T and F are the number of time frames and frequency bins, respectively. The time-domain loss is based on the mean square error (MSE) between the denoised speech and clean speech, which is defined as:

$$loss_T = \frac{1}{N}\sum_{i=0}^{N-1}(x_i - \hat{x}_i)^2 \quad (35)$$

where x and $\hat{x}$ are the samples of the clean speech and the denoised speech, respectively, and N denotes the number of samples. The final loss function combines the two types of losses mentioned above, namely:

$$loss = \mu * loss_F + (1 - \mu) loss_T \tag{36}$$

where μ is a tunable parameter and set as 0.4 in our work

2.4 Experimental results

2.4.1 Experimental settings

We carry out our experimental studies based on a public speech dataset [35] selected from Voice Bank corpus, including 14 female and 14 male speakers of 11572 utterances for training and one male and one female speakers of 824 utterances for testing. All utterances are sampled at 48 kHz. The noisy mixtures are formed for training by using 2 artificially generated noises plus 8 types of noises from DEMAND dataset [37] and setting SNR levels to 0 dB, 5 dB, 10 dB and 15 dB. The noisy speech for testing is generated from 5 types of unseen noises (café, bus, living, psquare, office) at SNR levels of 2.5 dB, 7.5 dB, 12.5 dB and 17.5 dB. We adopt several performance measures to evaluate our proposed speech enhancement model: perceptual evaluation of speech quality (PESQ) [1] with a score between -0.5 and 4.5; Short-time objective intelligibility (STOI) [40] with a score ranging from 0 to 1; We also adopt three subjective mean opinion scores (MOSs) valued between 1 and 5 [42] for more comprehensive evaluation, i.e., CSIG for signal distortion, CBAK for evaluating the intrusiveness of background noise and COVL for overall speech quality.

Two different inputs can be used for training SE systems, including speech waveform leading to time-domain structure and speech spectrogram resulting in T-F domain method. In our experiments, we mainly focus on the waveform inputs, where we adopt pre-processing method for processing long-rang speech waveform. We first resample all utterances to 16 kHz. For this scenario, we randomly select a 4-second segment from each utterance if that is longer than 4 seconds for reducing the computational time. Within each batch, the smaller utterances are zero-padded to make the equal length. In the to-chunk stage, we use the frame length of 512 samples

(32ms) with 50% overlap (256 samples) between adjacent frames, constructing a reduced sequence length based on the Eq. (33) introduced above. The feature channel of waveform inputs is 1 (C_{in} = 1). To have a fair comparison, we adopt the same model setting for proposed TSTNN and CPTNN except for inserted transformer blocks. More specifically, proposed TSTNN and CPTNN utilize 2 down-sampling layers in encoder and 2 up-sampling layers in decoder, where each layer includes a dilated-dense block using depth-wise separable convolution with dilation factor of 3. We use 6 TSTBs in proposed TSTNN and 4 CPTBs in proposed CPTNN, generating the same number of transformer blocks in two models. For proposed transformer setting, the input feature dimension is set as 32 and 4 attention heads are employed in MHA. Within the modified FFN block, we use a bidirectional GRU layer with 64 hidden units to track the position information of speech sequence. Other model configurations are maintained as described above chapter.

We train and optimize our proposed models for 100 epochs by using Adam algorithm [60]. To alleviate the gradient explosion problem in sequence models, the gradient clipping is used to constrain the L2 norm of the gradient to 5. We adopt the dynamic strategies for the learning rate during training [31, 60]. More specifically, within the first 4000 training steps, the learning rate is initially linearly increased from 0 to $4e^{-4}$. After that, the learning rate is decayed by 0.98 for every two epochs. It can be defined below:

$$Lr = \begin{cases} k_1 \cdot d^{-0.5} \cdot n \cdot n\_warmups^{-1.5}, & n \leq n\_warmups \\ k_2 \cdot 0.98^{\lfloor epoch/2 \rfloor}, & n > n\_warmups \end{cases} \quad (37)$$

where n denotes training steps, and $k_1 = 0.2$ and $k_2 = 4\text{e}^{-4}$ are hyperparameters. *n_warmups* denotes the number of warm-ups, which is set as 4000, d is the feature dimension of input, which is set to 64 for transformer blocks.

2.4.2 Comparisons with baselines

Table I gives the comparison results for the proposed model and some of the existing time-domain and T-F domain methods, evaluated on the same dataset [35]. First, our proposed TSTNN and CPTNN achieve an impressive performance compared with existing models. Especially, our CPTNN has a significant improvement on PESQ value from 1.97 to 3.07, showing superior performance with the lighter model complexity (only 0.76 million parameters)

than most of methods in time domain. Although CPTNN achieves the same PESQ score as DEMUCS (3.07) with large model configuration, CPTNN has only about 44 times fewer parameters than DEMUCS (33.5 million parameters) and it is trained without employing any data augmentation technologies. Compared with TSTNN using cascaded transformer structure, CPTNN further has a 0.1 PESQ improvement. In view of the STOI score, TSTNN and CPTNN have the highest STOI score (95%) compared with all other time-domain methods. Besides, CPTNN has superior performance in three MOSs evaluation metrics than TSTNN and other current waveform-based approaches.

Second, CPTNN achieves the state-of-the-art evaluation performance compared with all T-F domain methods in terms of the PESQ and STOI. Moreover, the PESQ value achieved by CPTNN is about 0.14 higher than that of DCUNet-16 using complex network. Although DCUNet-16 has a better score than CPTNN in CBAK, it involves more parameters (2.3 million) which is about 3 times larger than CPTNN (0.76 million parameters). Meanwhile, CPTNN achieves the slightly higher PESQ value than T-GSA which is a transformer with modified self-attention, and it has better performance in three MOSs evaluation scores.

By comparing with existing time-domain and T-F domain methods, our proposed TSTNN and CPTNN achieve a competitive performance in modeling long-range speech sequences while having relatively low model parameters. The main reason is that our proposed local and global transformer structure can extract abundant local and global information, which are important for long-range speech sequences. When we further conduct the comparison between proposed TSTNN and CPTNN, the better performance of speech enhancement is achieved by the CPTNN. This finding makes us believe that our proposed cross-parallel transformer architecture not only efficiently extracts local and global contextual information, but also adopts an effective fusion block based on proposed cross-attention to fuse the extracted local and global features, which can promote the ability of the model to learn the long-range dependencies of speech sequences.

2.4.3 Selection of μ in loss function

To evaluate the effect of μ in loss function, we set its value as {0.0, 0.2, 0.4, 0.6, 0.8, 1.0} to observe the performance with proposed CPTNN structure. As shown in Fig. 2.20 and Fig. 2.21, when μ is 0.4, we can achieve best PESQ and STOI values. For μ = 1.0, i.e., when the frequency

Table 1: Experimental results of the proposed models and existing models on the same dataset [35]. Five evaluation metrics are used (higher values, better performance) and the number of trainable parameters is considered ('T-F' and 'T' denotes the T-F domain and time domain)

Model	T/T-F	PESQ	STOI	CSIG	CBAK	COVL	Para.
Noisy	-	1.97	91	3.34	2.44	2.63	-
Wiener	-	2.22	-	3.23	2.68	2.67	-
SEGAN, [61]	T	2.16	93	2.48	2.94	2.80	97.47
Wavenet, [62]	T	-	-	3.62	3.23	2.98	-
Wave U-Net, [23]	T	2.40	-	3.52	3.24	2.96	10.0
Attention Wave U-Net, [63]	T	2.62	-	3.91	3.35	3.27	-
MetricGAN, [64]	T-F	2.86	-	3.99	3.18	3.42	-
DCUNet-16, [65]	T-F	2.93	-	4.10	**3.77**	3.52	2.3
PHASEN, [66]	T-F	2.99	-	4.21	3.55	3.62	-
Self-Adapt MHSA, [67]	T-F	2.99	-	4.15	3.46	3.51	-
Low-latency U-Net, [68]	T-F	2.90	94	4.22	3.32	3.58	-
NAAGN, [69]	T-F	2.90	-	4.13	3.50	3.51	-
DEMUCS, [27]	T	**3.07**	**95**	4.31	3.4	3.63	33.5
T-GSA, [49]	T-F	3.06	-	4.18	3.59	3.62	63
CAUNet, [70]	T	2.96	**95**	4.22	3.53	3.60	1.04
CARN, [71]	T-F	2.93	**95**	4.19	3.61	3.54	-
SA-TCN, [72]	T-F	3.02	94	4.29	3.50	3.67	9.91
TSTNN (ours)	T	2.96	**95**	4.33	3.53	3.67	0.92
CPTNN (ours)	T	**3.07**	**95**	**4.40**	3.59	**3.76**	**0.76**

loss is exclusively used, the PESQ and STOI values are the worst, showing that using frequency loss alone as objective function, the performance will be lower. When we slightly decrease μ from 1.0 to 0.4, the PESQ score raises from 2.16 to the best one 3.07. However, when we continue to decrease μ from 0.4 to 0.0, the PESQ value shows the decreasing trend. For STOI, the scores are similar when μ varies from 0.2 to 0.6 but are worse when μ is 0.8 or 1.0. This observation above shows that using only time-domain or frequency-domain loss as training target would yield a worse performance of speech denoising. In other words, a combined frequency and time domain loss function with a suitable weight will make the denoising performance much better.

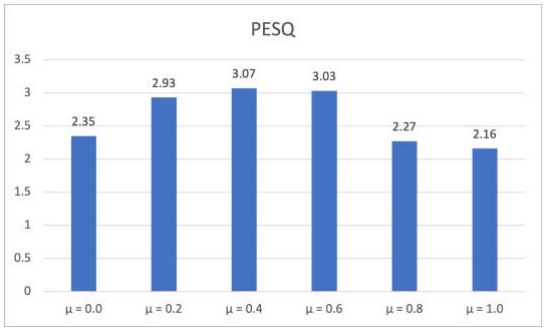


Figure 2.20: PESQ scores based on various μ

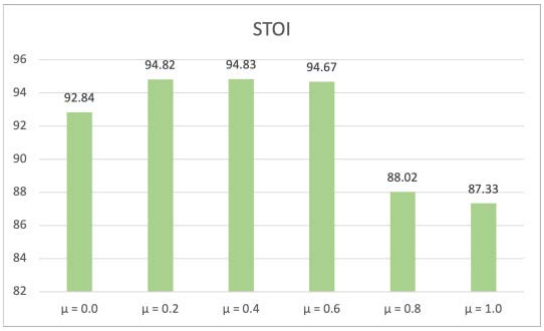


Figure 2.21: STOI scores based on various μ

2.4.4 Ablation study

In this section, we present some findings of ablation study performed to demonstrate the effectiveness of different components of our model. To simplify the experiments, we only conduct all comparison experiments on proposed CPTNN structure. First, inspired by the effectiveness of RNNs in keeping track of the positional information, we replace the first FC layer of FFN of vanilla transformer by a GRU layer to learn the positional information of speech sequences. In order to further explore the importance of positional information and the superiority of our modified FFN, we here set two comparison experiments by using the standard FFN of vanilla transformer with or without positional encoding (PE) module of vanilla transformer in our proposed model.

Table 2: Evaluation results among different configurations with positional encoding and FFN

Model	PESQ	STOI (%)	CSIG	CBAK	COVL
CPTNN (GRU-FFN, ours)	**3.07**	**95**	**4.40**	**3.59**	**3.76**
CPTNN (Vanilla FFN + PE)	2.77	94	4.20	3.41	3.50
CPTNN (Vanilla FFN)	2.71	94	4.15	3.38	3.46

As shown in Table 2, the model using only standard FFN without PE module has the worst performance, while the one incorporating PE module achieves a better performance, indicating that the positional information is very important for improving the performance of speech enhancement. By implementing our modified FFN without PE module, there is an obvious

improvement on denoising performance. The possible reason is that the PE of vanilla transformer does not work well for acoustic sequences. Specifically, for speech denoising models, the input features are fixed vectors rather than trainable word embeddings, making it difficult for the model to separate position and content for each state. Instead, in our modified FFN module, we use RNN-based layer to effectively learn the positional information during the training.

Table 3: Network configurations with different fusion methods

Model	Fusion	Block	Para.(M)
CPTNN (ours)	√	4 CPTBs	0.76
APTNN	√	6 APTBs	0.76
PTNN	×	6 PTBs	0.76

Second, the cross-parallel transformer block (CPTB) is designed to extract local and global features by using local and global transformers in parallel, followed by a cross transformer acting as a fusion block to learn the interaction between the local and global information during training, leading to a better fused feature representation. In order to further explore the influence of fusion mechanism in CPTB on evaluation performance, we here design two different transformer blocks both modified from our proposed CPTB, where one exclusively replaces the cross transformer by average operation named as average parallel transformer block (APTB), and the other simply removes the cross transformer named as parallel transformer block (PTB). To make a fair comparison, we use 6 APTBs and 6 PTBs to separately replace the transformer blocks of our proposed CPTNN, generating two reference models whose trainable parameters are same as the CPTNN, namely APTNN and PTNN as shown in Table 3, respectively.

Table 4: Evaluation results from various comparison models with different fusion methods

Model	PESQ	STOI (%)	CSIG	CBAK	COVL
CPTNN (ours)	**3.07**	**95**	**4.40**	**3.59**	**3.76**
APTNN	3.02	94	4.35	3.48	3.71
PTNN	2.91	93	4.28	3.49	3.61

As shown in Table 4, our proposed CPTNN reaches the best performance in all evaluation metrics compared with other reference models. Especially, the PTNN that has only parallel transformer blocks obtains the worst performance, showing that the interaction between local and global information is important for improving denoising performance. The APTNN that simply uses average operation as the fusion block has better performance than PTNN, but it is still inferior to the proposed CPTNN with cross-attention based fusion block, indicating that our proposed cross-transformer can efficiently fuse the local and global information by learning their interactions during training to obtain a better contextual feature representation of long-range speech sequences.

Table 5: Evaluation results among different structures in transformer block

Model	PESQ	STOI (%)	CSIG	CBAK	COVL
CPTNN (Cross-Parallel, ours)	**3.07**	**95**	**4.40**	**3.59**	**3.76**
CPTNN (Cascaded)	3.03	**95**	4.40	3.55	3.75
CPTNN (Parallel)	2.91	93	4.28	3.49	3.61

Next, we explore the effectiveness of our cross-parallel structure by comparing it with cascaded structure and parallel structure and provide relevant scores in Table 5. For the parallel structure, we simply employ the parallel transformer block (PTB) mentioned above by removing cross-attention transformer in CPTB. For the reference model with cascaded structure, we adopt a similar transformer structure implemented by TSTNN. In TSTNN, a two-stage transformer block was proposed to effectively extract contextual information, where the local transformer and global transformer are applied to extract local and global information with cascaded connection iteratively. In order to verify the effectiveness of our proposed cross-parallel structure, we only replace the cross-parallel structure of the transformer block with a cascaded structure just like TSTNN, where six cascaded transformer blocks are employed to construct a reference model with the same number of trainable parameters as CPTNN for a fair comparison.

As shown in Table 5, the parallel architecture obtains the worst speech denoising performance especially PESQ and STOI scores when compared with other two structures. The possible reason is that the lack of communications between local and global information impedes

the contextual modeling of long-range speech sequences. For the cascaded structure, it reaches the same value in STOI and CSIG as the cross-parallel architecture while performs worse in other evaluation metrics due to the fact that the inputs of global transformer in the sequential structure are far from the inputs of local transformer. Instead, our proposed CPTNN extracts the local and global information simultaneously by utilizing local and global transformers, and their outputs are fused by the efficient cross-attention based transformer.

Table 6: Evaluation results among two different cross-attention mechanisms

Model	PESQ	STOI (%)	CSIG	CBAK	COVL
CPTNN (CrossAtt_V1, ours)	**3.07**	**95**	**4.40**	**3.59**	**3.76**
CPTNN (CrossAtt_V2)	2.94	**95**	4.34	3.52	3.66

Finally, we further analyze the effectiveness of the cross-attention based transformer which fuses the two output features from local and global transformer respectively, where one output represents the information of local dependencies within each frame and the other one means a sum of global dependencies of different frames. In our proposed cross-attention transformer, the keys (K) and values (V) are from the features processed by the global transformer, and queries (Q) are the outputs from the local transformer. This allows the global transformer to receive the contributions from the local transformer for generating the fused contextual feature representation.

Actually, there is another choice to make the outputs of the local transformer act as keys (K) and values (V) and the outputs from the global transformer perform as queries (Q). For comparing these two options, we design a comparison model which implements the cross-attention mechanism to make the global transformer contribute on the local transformer for completing the fusion, namely CrossAtt_V2. From Table 6, we can observe that the comparison model based on CrossAtt_V2 mechanism has a 0.12 reduction in PESQ score and worse results in CSIG, CBAK and COVL. It indicates that our proposed cross-attention mechanism is more efficient for fusing the local and global information as it keeps the global transformer absorb the supplements from the local transformer.

2.4.5 Performance on selective noise with different SNRs

Here we investigate our proposed model and other configurations on subset of testing data, where challenging Café and Psquare noises are used to generate the noisy utterances at SNRs of 2.5 dB, 7.5 dB, 12.5 dB and 17.5 dB. As shown in Table 7, our proposed CPTNN (μ = 0.4) has made significant improvement on the denoising performance in Café and Psquare noises, in term of PESQ and STOI scores. For PESQ, our proposed CPTNN achieves obvious improvement of the score in two given noises with all given SNR levels. More specifically, proposed CPTNN has best PESQ values in Café noise environment at all four SNRs among all configurations. Meanwhile, CPTNN obtains best PESQ improvement in Psquare noise environment at 7.5 dB and 17.5 dB and has slightly lower score than CPTNN (μ = 0.6) and cascaded structure at SNRs of 2.5 dB and 12.5 dB, respectively. For STOI, the proposed CPTNN has the best score in Psquare noise at most of SNR levels while has the slightly lower score than CPTNN (μ = 0.2) and CPTNN with CrossAtt_V2 mechanism in Café noise case. By comparing with other configurations, the CPTNN indicates the superiority of speech denoising at challenging noise environment of low SNR level.

Table 7: PESQ and STOI comparisons among different configurations

Metric	PESQ										STOI									
Noise Type	Café					Psquare					Café					Psquare				
Noise SNR (dB)	2.5	7.5	12.5	17.5	Avg.	2.5	7.5	12.5	17.5	Avg.	2.5	7.5	12.5	17.5	Avg.	2.5	7.5	12.5	17.5	Avg.
Unprocessed	1.15	1.33	1.54	1.95	1.49	1.24	1.46	1.84	2.39	1.73	78.1	87.3	91.2	94.4	87.8	86.9	91.3	93.8	93.8	91.4
CPTNN (ours, μ=0.4)	**1.93**	**2.49**	**2.88**	**3.21**	**2.63**	2.38	**2.81**	3.04	**3.39**	**2.91**	88.1	92.5	94.6	95.4	92.6	**91.5**	**94.8**	**95.8**	94.9	**94.2**
CPTNN, μ=0.0	1.57	1.91	2.16	2.52	2.04	1.89	2.12	2.37	2.66	2.26	85.3	90.6	92.5	94.4	90.7	89.5	92.9	94.3	93.2	92.5
CPTNN, μ=0.2	1.85	2.42	2.70	3.03	2.50	2.37	2.69	2.92	3.24	2.81	88.3	92.5	**96.6**	**95.6**	**93.3**	91.2	**94.8**	95.7	**95.1**	**94.2**
CPTNN, μ=0.6	1.92	2.48	2.82	3.12	2.59	**2.45**	**2.81**	3.02	3.35	**2.91**	87.9	92.2	94.4	**95.6**	92.5	91.2	**94.8**	**95.8**	**95.1**	**94.2**
CPTNN, μ=0.8	1.57	2.01	2.21	2.36	2.04	1.98	2.17	2.31	2.45	2.23	82.0	86.1	87.7	88.9	86.2	85.8	88.2	89.0	88.0	87.8
CPTNN, μ=1.0	1.61	1.89	2.12	2.24	1.97	1.91	2.11	2.18	2.33	2.13	81.6	85.5	87.0	88.2	85.6	85.4	87.2	88.3	87.5	87.1
CPTNN, Vinalla FFN + PE	1.75	2.23	2.52	2.98	2.37	2.15	2.53	2.82	3.19	2.67	86.7	91.3	93.6	95.1	91.7	90.7	93.7	95.4	94.4	93.6
CPTNN, Vinalla FFN	1.65	2.11	2.44	2.87	2.27	2.03	2.45	2.71	3.15	2.59	85.7	90.7	93.1	95.2	91.2	89.7	93.7	95.2	94.5	93.3
PTNN	1.81	2.38	2.76	3.14	2.52	2.34	2.75	3.03	3.37	2.87	86.7	91.9	94.1	95.4	92.0	91.1	94.4	95.5	94.9	94.0
APTNN	1.82	2.43	2.87	3.18	2.58	2.40	2.80	3.04	**3.39**	**2.91**	87.1	92.0	94.5	**95.6**	92.3	91.4	94.7	95.6	95.0	**94.2**
Cascaded	1.89	2.47	2.81	3.16	2.58	2.39	2.73	**3.06**	3.34	2.88	87.3	92.1	94.4	**95.6**	92.4	**91.5**	94.6	95.7	95.0	**94.2**
CPTNN, CrossAtt_V2	**1.93**	**2.49**	2.74	3.04	2.55	2.40	2.73	2.92	3.18	2.81	**88.5**	**92.8**	94.8	**95.6**	92.9	**91.5**	**94.8**	**95.8**	94.8	**94.2**

Moreover, we present the enhanced spectrogram of proposed CPTNN and other models with different configurations in a Café noisy environment with a SNR level of 2.5 dB. As shown in Fig. 2.22, we first found that proposed CPTNN has better denoising performance than most of comparison models in terms of highlighted regions (red and yellow regions). Second, proposed

CPTNN has a similar harmonic structure as Cascaded model, but the proposed CPTNN could restore more approximate structure than Cascaded model in highlighted region (orange region). This proves that proposed CPTNN is more efficient to reconstruct the spectrum details and make them be similar as the clean speech in the challenging noise conditions.

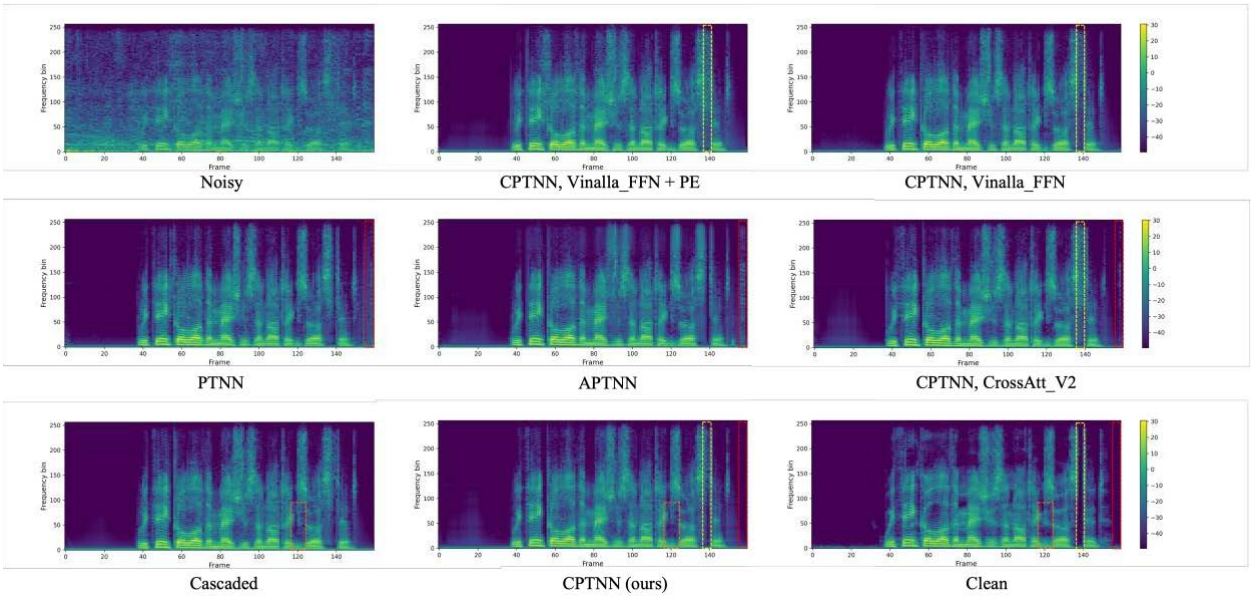


Figure 2.22: Spectrogram of different configurations based on an audio sample mixed with Café noise at an SNR of 2.5 dB

2.5 Summary

In this chapter, we have proposed two transformer neural networks for speech enhancement, including two-stage transformer neural network (TSTNN) and cross-parallel transformer neural network (CPTNN). We first modified the standard transformer encoder by removing the positional encoding and replacing the FFN by a GRU-augmented FFN for positional information tracking. The TSTNN and CPTNN are built based on a similar framework such as encoder, masking module and decoder, except for the transformer blocks. The TSTNN adopts several pairs of local and global transformers with cascaded structure to extract the long-term dependency of speech sequence. The local transformer performs on each frame to learn the local-term information. The global transformer is implemented across multiple frames to capture the long-term information. To overcome the potential drawback of information leakage caused by the cascaded structure, our proposed CPTNN employs multiple cross-parallel transformers to

parallelly extract local and global information, whose outputs are then fused by the cross-attention transformer to obtain the adaptive contextual information. Note that, the obtained contextual information from two-stage transformers or cross-parallel transformer will pass through a masking module to create a mask which will be used to filter the encoder outputs for reconstructing the enhanced speech via a decoder.

The extensive experimental results indicate that our proposed TSTNN and CPTNN have achieved an impressive performance of speech enhancement compared with existing methods in all evaluation metrics. Especially CPTNN obtains a better performance than TSTNN, which shows the effectiveness of cross-parallel structure on long-range dependency modeling. Furthermore, our proposed TSTNN and CPTNN involve relatively fewer trainable parameters than current approaches.

Chapter 3

Attention-free neural networks for speech enhancement

In this chapter, we propose a novel and attention-free architecture based on multi-layer perceptron (MLP) for speech enhancement, named SE-Mixer. This chapter is organized as follows. We first overview the MLP-Mixer in Section 3.1. Based on this, the recent works using Mixer structure for speech tasks are introduced. In Section 3.2, we describe the proposed SE-Mixer and single-channel speech enhancement. And Section 3.3 presents the evaluation performance of the proposed SE-Mixer.

3.1 Introduction to MLP-Mixer

Transformer-based neural networks have achieved an impressive success in many fields including NLP and CV. However, for long-term sequences, the transformer is very computation-expensive due to numerous matrix operations. Recently, many researchers have studied attention-free models and achieved competitive performance to transformer-based models but at reduced model complexity. The MLP-Mixer model is proposed in [73] for exploring an MLP based structure for image classification without attention operation. Different from the transformer using multi-head self-attention to capture the long-term relationship among the visual features, the MLP-Mixer is a simple architecture only replying on the MLP structure, basic matrix multiplication operation and changes of data layout. As shown in Fig. 3.1, the proposed MLP-Mixer is mainly based on the Mixer layers and FC layers. As a core idea, the Mixer layer is comprised of one token-mixer MLP and one channel-mixing MLP, each containing two FC layers and a GELU non-linearity in between as indicated in Fig. 3.2.

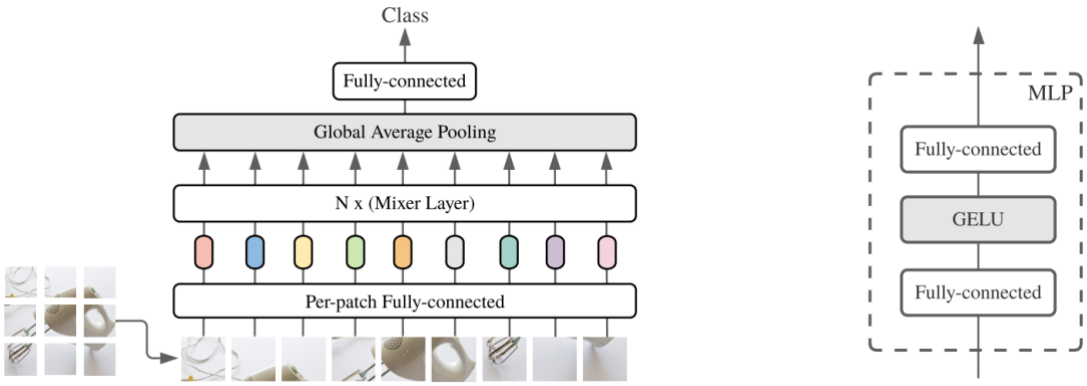


Figure 3.1: MLP-Mixer for image classification [73]

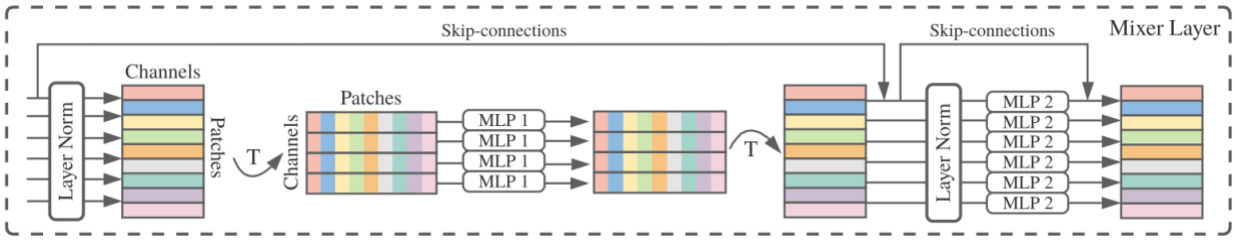


Figure 3.2: The details of Mixer layer [73]

In the proposed MLP-Mixer, the Mixer layer receives the sequence of S image patches which are obtained by splitting the whole image without overlaps. Each image patch is projected to form the patch with feature dimension C. The sequence of image patches will form the two-dimensional input data $X \in \mathbb{R}^{S \times C}$, where $S = HW/P^2$ is the number of image patches, H and W denote the height and width of the whole image, P means the resolution of image patch. The input X is first passed through the channel-mixing MLP to make different channel features communicate with each other, which implements on the dimension C. Then, the token-mixing MLP is employed to act on the dimension S of input X for building the connection among different locations or tokens, where the locations and tokens mean the image patches. Additionally, the layer normalization is used before each MLP, and skip-connection is applied after each MLP. The whole procedures can be formulated as follows:

$$Out_{channel-mixing} = X + Reshape(W^2\sigma(W^1(Reshape(LN\ (X))[:, i])))) \quad (38)$$

$$Out_{token-mixing} = Out_{channel-mixing} + W^3\sigma(W^4 LN\ (Out_{channel-mixing})[j, :])) \quad (39)$$

where $i = 1 \ldots C$ and $j = 1 \ldots S$ are the index of channel and that of the token dimension, W^1

and W^2 are the linear transformation matrix of channel-mixing MLP, W^3 and W^4 are the linear transformation matrix of token-mixing MLP, and $\sigma(\cdot)$ is the operation of GELU non-linearity.

Inspired by the MLP-Mixer proposed for image classification as an attention-free model, authors of [74] adopted the mixer-based structure for speech synthesis where both time mixing and channel mixing are proposed for successively extracting the temporal information and spatial information of speech features as shown in Fig. 3.3. More specifically, in time mixing, to overcome different input sequence lengths, the depth-wise 1-D convolutional layer is adopted to replace the FC layer in the original MLP, allowing communication between the features of different time frames within a limited region. Then, the channel mixing employs the MLP to learn the relationship between different feature channels within a certain time frame, which operates on the dimension of feature channel. By using multiple stacked mixer blocks, the temporal and spatial information of speech sequences are learnt to form the contextual features for following procedures.

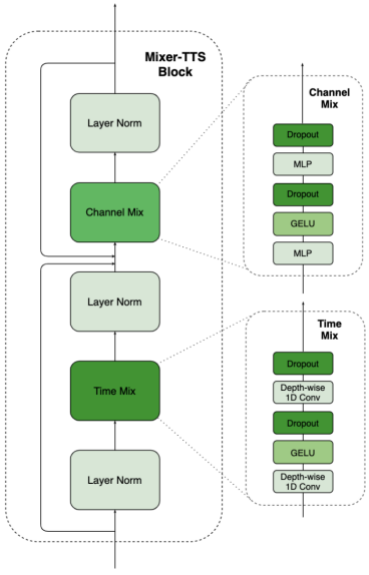


Figure 3.3: Mixer-TTS for text-to-speech task [74]

3.2 Proposed attention-free neural network

Many competitive denoising methods are realized by networks involving attention mechanism or transformer as a part of them but it is not thoroughly studied as to whether the attention is indispensable for resolving long-range dependency problems in speech enhancement, especially

when considering the trade-off between computational efficiency and denoising performance. Recently, a simple but efficient architecture named MLP-Mixer has been proposed in [73] for image recognition by exclusively using two FC layers as backbone to perform token mixing and channel mixing, exhibiting a comparable performance to transformer-based methods with lower computational complexity. Inspired by this study, in this section, we propose an MLP based architecture without attention mechanism for speech enhancement, named as SE-Mixer. We first present the whole structure of the proposed SE-Mixer and then describe our temporal MLP and frequency MLP for extracting temporal and frequency information of speech sequences.

3.2.1 Proposed SE-Mixer architecture

Fig. 3.4 shows the whole structure of our proposed SE-Mixer, which consists of an encoder, a mixing module and a decoder.

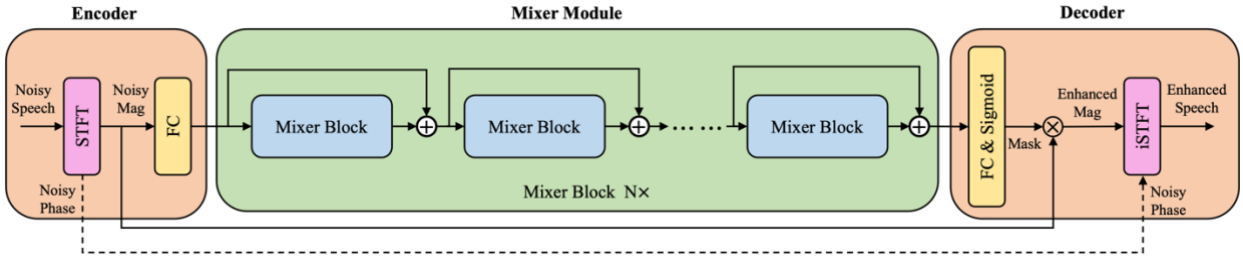


Figure 3.4: Overview of SE-Mixer architecture

The encoder consists of a STFT feature extractor and a FC layer to obtain the feature representation of noisy speech sequences. First, the magnitude and phase feature of noisy speech are extracted by using STFT, where the magnitude spectrum $M \in \mathbb{R}^{T\times F}$ is utilized as input feature, and the phase part $P \in \mathbb{R}^{T\times F}$is retained for final reconstruction of the enhanced speech waveform. Here T denotes the sequence length or the number of frames, and the F is the frequency dimension or the number of frequency bins. Then, a FC layer linearly projects the magnitude spectra into feature representations with reduced feature dimension $X \in \mathbb{R}^{T\times H}$, which will be processed by the following mixer module.

The mixer module is comprised of L mixer blocks, as shown in Fig. 3.4, to extract the contextual information of speech features X, where each one includes a temporal Conv-MLP and

a frequency MLP to iteratively extract temporal information from different time steps and frequency information at each time step. Moreover, sum of the output of each mixer block and residual connection from its input are passed through the next mixer block, which could avoid the gradient vanishing problem during the training. After the mixer module, the outputs $S \in \mathbb{R}^{T \times H}$ will have the same dimension as the input features X.

In the decoder, the output S from mixer module is first processed by a FC layer to increase the feature dimension from H to F, followed by the sigmoid non-linearity to obtain a mask $M_{mask} \in \mathbb{R}^{T \times F}$ with a value ranging from 0 to 1. The generated mask will be multiplied with the magnitude feature of the noisy speech in element-wise manner, creating the enhanced magnitude spectrum of speech. Then, the enhanced magnitude is employed along with the noisy phase to reconstruct the enhanced speech waveform by iSTFT.

3.2.2 Proposed temporal MLP and frequency MLP

In the MLP-Mixer proposed for image classification, the channel MLP and token MLP are proposed to perform the channel mixing and token mixing, where the channel information within each image token and spatial information of different tokens are learned for better feature representation. The MLP-Mixer, as an attention-free structure, indicates that a simple MLP-based structure can achieve a comparable performance to vision transformer neural networks with highly computational attention mechanism. Inspired by this study, we propose two MLPs for speech features to extract temporal information and frequency information, named as temporal MLP and frequency MLP.

Our proposed temporal MLP is designed to extract multi-scale temporal features from time steps since some continuous time frames have abundant speaker information, but some others are silent voice. The temporal MLP consists of a multi-scale convolution block, a FC layer and a GELU non-linearity in between, as shown in Fig. 3.5. It acts on time steps of inputs to perform temporal mixing. The multi-scale convolution block is utilized to extract rich temporal features over various time scales as shown in Fig. 3.6, where the input feature $X \in \mathbb{R}^{T \times H}$ is first processed by group normalization and then passes through a couple of bottleneck MLPs augmented by the dilated convolutions (T denotes time steps and H is feature dimension). The dilated convolution is useful for enlarging the receptive field of feature [25] and is applied in

time dimension of input features with an objective to control the different time scales. The bottleneck structure can decrease the redundancy of features by reducing the feature dimension with the first layer and then reconstructing the compressed features of the same dimension as the inputs with the second layer, which can involve the reduced model parameters.

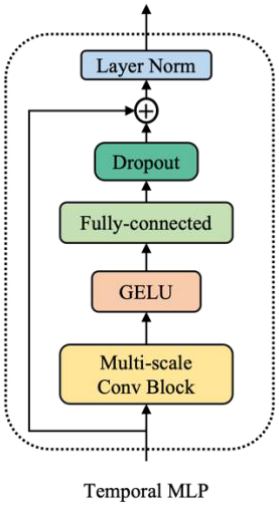


Figure 3.5: Proposed temporal MLP

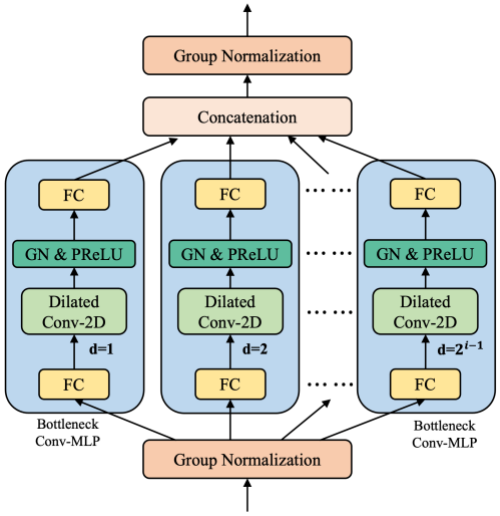


Figure 3.6: Proposed multi-scale convolution block

Within each bottleneck convolution-augmented MLP, a FC layer first decreases the feature dimension of inputs into C as shown in Eq. (40), where C is the number of hidden units. Next, the features are reshaped into $[C, 1, T]$ format and then fed into the dilated 2D convolution with kernels of size (1, 3) to learn the temporal information along the dimension T as given in Eq. (41), where dilation rate d controls the time scale. Next, the output features are reshaped back to $[T, C]$ format, which is followed by the group normalization and PReLU non-linearity, and then mapped by a FC layer to produce the final output of each scale as in Eq. (42). The outputs from all scales are concatenated together along feature channel and normalized by group normalization to obtain the final output of the multi-scale convolution block as shown in Eq. (43).

$$Proj_i = GN(X)W_i^{\alpha} + b_i^{\alpha} \tag{40}$$

$$Mid_i = Conv2D_i(Proj_i; k; d_i; p_i) \tag{41}$$

$$Scale_i = PReLU\big(GN(Mid_i)\big)W_i^{\beta} + b_i^{\beta} \tag{42}$$

$$MultiScale = GN\big(Concat(Scale_1, \cdots, Scale_i)\big) \tag{43}$$

where $W_i^{\alpha} \in \mathbb{R}^{H \times C}$, $b_i^{\alpha} \in \mathbb{R}^{C}$, $W_i^{\beta} \in \mathbb{R}^{C \times S}$, and $b_i^{\beta} \in \mathbb{R}^{S}$ denote linear transformation matrices

of the ith bottleneck block, i=1, 2, $\cdots$, h, h being the number of bottleneck blocks, and S is the hidden units of the second FC layer. In Eq. (41), k is the kernel size of 2D convolution which is set as (1, 3), $d_i = 2^{i-1}$ means the dilation rate in ith bottleneck block, and p_i is the padding for convolution and set as $2 \times d_i$. $GN(\cdot)$ and $PReLU(\cdot)$ are the group normalization and non-linearity operation, respectively.

Then, the output from multi-scale convolution block is processed by the GELU non-linearity and a FC layer to obtain the final output of temporal Conv-MLP, where residual connection and layer normalization are applied as well, which can be denoted as follows:

$$Output_T = GELU(MultiScale)W^Y + b^Y \tag{44}$$

$$MLP_T = LN(Output_T + X) \tag{45}$$

where $W^Y \in \mathbb{R}^{d_T \times H}$ with $d_T = h \times S$, $b^Y \in \mathbb{R}^H$ denotes the linear transformation matrix for output of temporal Conv-MLP. $LN(\cdot)$ is layer normalization operation.

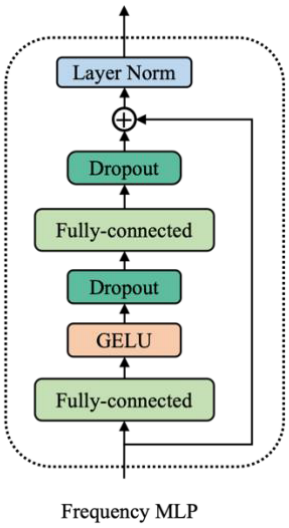


Figure 3.7: Proposed frequency MLP

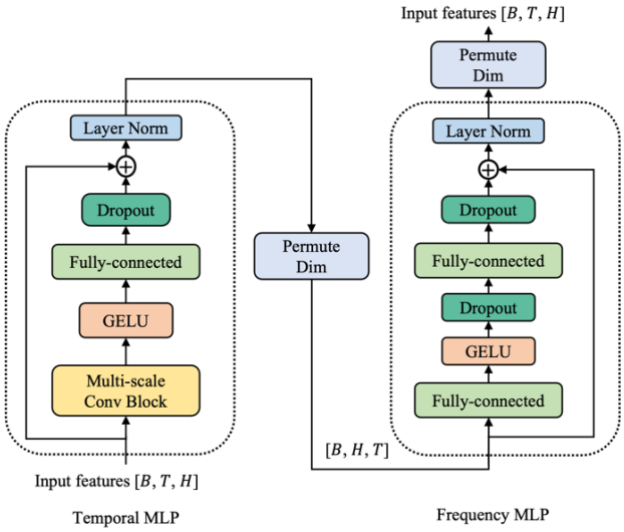


Figure 3.8: Proposed mixer block

After extracting temporal features, the frequency MLP is proposed to extract frequency information at each time step by using a pure MLP including two FC layers, as shown in Fig. 3.7, which acts on frequency channel to perform frequency mixing. The GELU non-linearity and dropout operation are also used to promote the generalization and avoid overfitting. In addition, the residual connection and layer normalization are adopted to obtain the final outputs of the frequency MLP. The procedures are defined as follows:

$$Output_F(MLP_T) = GELU(MLP_T W^{\delta} + b^{\delta}) W^{\varepsilon} + b^{\varepsilon} \tag{46}$$

$$MLP_F = LN(Output_F + MLP_T) \tag{47}$$

where $W^{\delta} \in \mathbb{R}^{H \times d}$, $b^{\delta} \in \mathbb{R}^{d}$, $W^{\varepsilon} \in \mathbb{R}^{d \times H}$ and $b^{\varepsilon} \in \mathbb{R}^{H}$ denote linear transformation matrices of two FC layers. H is the feature dimension of outputs from temporal Conv-MLP and d is the hidden units of the first FC layer. Note that our frequency MLP utilizes the bottleneck structure to generate the dimension reduction in the intermediate layer by setting d as $H/4$.

Based on the proposed temporal MLP and frequency MLP, we constitute a mixer block to successively extract temporal information from different time steps and frequency information at each time step. As shown in Fig. 3.8, the input features with temporal dimension T and frequency dimension F are first passed through the temporal MLP to extract temporal information with different time scales, generating multi-scale temporal features for subsequent frequency processing. The obtained temporal features are first processed by permutation block to get the data format with transposed dimension, which is then fed to the frequency MLP to extract frequency-wise features at each time step. Finally, the output features are temporal-wise and frequency-wise feature representations, which are permuted to form the same dimension as the inputs of the mixer block.

3.2.3 Loss function

We use power-compressed MSE loss [66] to train our model, including amplitude loss and phase-aware loss for considering both magnitude and phase information. The amplitude loss is defined as follows:

$$\begin{aligned} \mathcal{L}_{amplitude} &= MSE\left(abs\left(\hat{X}_{cprs}^{enh}\right), abs\left(X_{cprs}^{cln}\right)\right) \\ &= \frac{1}{TF} \sum_{t=0}^{T-1} \sum_{f=0}^{F-1} (\left|\hat{X}(t,f)\right|^{p} - |X(t,f)|^{p})^2 \end{aligned} \tag{48}$$

where $\hat{X}_{cprs}^{enh}$ and X_{cprs}^{cln} are the power-law compressed spectrogram of enhanced speech spectrogram and clean speech spectrogram, respectively, $abs(\cdot)$ is the operator for obtaining the absolute value, T and F are the number of time frames and frequency bins, p is the compression factor, which is set as 0.3. Different from the amplitude loss, the phase-aware loss is computed based on the MSE loss between the enhanced speech spectrogram and the clean spectrogram with a power-law compression, which is formulated as follows:

$$\mathcal{L}_{phase} = MSE\left(\hat{X}_{cprs}^{enh}, X_{cprs}^{cln}\right)$$
$$= \frac{1}{TF}\sum_{t=0}^{T-1}\sum_{f=0}^{F-1}(\hat{X}(t,f)^p - X(t,f)^p)^2 \quad (49)$$

where $X^p = |X|^p e^{j\angle X}$, and p is the compression factor, which is set as 0.3 in our experiments. Our final loss function is a weighed sum of amplitude loss and phase-aware loss, which is defined as follows:

$$\mathcal{L}_{PCMSE} = \alpha\mathcal{L}_{amplitude} + \beta\mathcal{L}_{phase}$$
$$= \frac{1}{TF}\sum_{t=0}^{T-1}\sum_{f=0}^{F-1}[\alpha\left(\left|\hat{X}(t,f)\right|^{0.3} - |X(t,f)|^{0.3}\right)^2 +$$
$$\beta(\hat{X}(t,f)^p - X(t,f)^p)^2] \quad (50)$$

where α and β are hyperparameters set as α =10, β = 1 in our work. Instead of only considering the pure amplitude information or pure phase information, both phase and amplitude information are taken into consideration in the loss function, which makes the phase in each T-F bin with higher amplitude to be emphasized. In other words, our loss function can supervise the model to focus on the T-F bins with high amplitude where abundant speech signals are located.

3.3 Experimental results

3.3.1 Experimental setups

To compare the proposed SE-Mixer with other existing methods, we implement our experiments on a publicly available VCTK dataset, including 28 speakers of 11572 utterances for training set and 2 speakers of 824 utterances for testing set. For training, the noisy samples are mixed with 2 types of artificially generated noises and 8 types of noises from the DEMAND at 4 signal-to-noise ratio (SNR) values of 0, 5, 10, and 15 dB. The testing dataset is synthesized by 5 unseen noises from the DEMAND with 4 SNRs of 2.5, 7.5, 12.5 and 17.5 dB.

All the utterances are resampled to 16 kHz and the 512-point STFT is applied to each 3-second segment by using 30ms Hann window with 10ms stride, leading to a 257-D feature vector in each frame. In the proposed SE-Mixer, the FC layer of encoder and decoder has 128 and 257 hidden units, respectively. For the mixer module, we use 8 proposed mixer blocks, each employing 4 bottleneck Conv-MLP blocks with exponentially increasing dilation rate (1, 2, 4

and 8), leading to four different time scales. In each bottleneck Conv-MLP block, 32 and 64 hidden units are used by the first and final FC layer. During the training stage, the proposed SE-Mixer is trained for 300 epochs and optimized by Adam algorithm [59]. Moreover, we dynamically schedule the learning rate while training. More specifically, the initial learning rate is set as 0.01 and then is decayed to $1e^{-5}$ by using Cosine annealing schedule [75].

We evaluate our proposed model by adopting five evaluation metrics, including two objective metrics and three subjective mean opinion scores (MOSs). The perceptual evaluation of speech quality (PESQ) is a popular objective metric to measure the enhanced speech, with a value between -0.5 and 4.5. Another metric short-time objective intelligibility (STOI) ranges from 0 to 1. The three MOSs are CSIG evaluating speech distortion, CBAK reflecting noise distortion and COVL measuring for overall speech quality. All these scores have a value ranging from 1 to 5.

3.3.2 Comparisons with baselines

Table 8 summarizes the comparison results of our SE-Mixer model with some of the state-of-the-art speech denoising models on the same dataset [35]. First, SE-Mixer obviously improves the PESQ value from 1.97 to 3.05, which shows a superior performance to most existing time-domain and T-F domain models while at the lowest model complexity (only 0.71 million parameters). In view of speech intelligibility performance, SE-Mixer obtains the best STOI score (95%) among the comparison methods. SE-Mixer obtains a slightly inferior PESQ value (3.05) than DEMUCS (3.07), but it has only about 47 times fewer trainable parameters than DEMUCS (33.5 million parameters) and moreover it is implemented without any data augmentation technologies. Besides, SE-Mixer has comparable achievement on MOSs to current models, where the DCUNet-16 achieves the best CBAK score using complex-valued network but worse performance than SE-Mixer in other metrics.

Second, we further compare the proposed SE-Mixer with some attention-based and transformer-based models to verify whether the attention mechanism is indispensable for processing long-range speech sequences in speech enhancement. From Table 8, we can observe that our attention-free SE-Mixer still achieves comparable and even superior performance to attention-based denoising models. Specifically, SE-Mixer has a better PESQ improvement than Attention Wave-UNet, NAAGN, Self-Adapt MHSA, CARN, SADNUNet and TSTNN, but

Table 8: Experimental results of the proposed and existing models on the VCTK dataset

Model	T/T-F	PESQ	STOI	CSIG	CBAK	COVL	Para.	MACs
Noisy	-	1.97	91	3.34	2.44	2.63	-	-
Wiener	-	2.22	-	3.23	2.68	2.67	-	-
SEGAN, [61]	T	2.16	93	3.48	2.94	2.80	97.47	9.47
Wavenet, [62]	T	-	-	3.62	3.23	2.98	-	-
Wave U-Net, [23]	T	2.40	-	3.52	3.24	2.96	10.0	2.45
Attention WaveUNet [63]	T	2.62	-	3.91	3.35	3.27	-	-
MetricGAN, [64]	T-F	2.86	-	3.99	3.18	3.42	-	-
DCUNet-16, [65]	T-F	2.93	-	4.10	**3.77**	3.52	2.3	21.01
PHASEN, [66]	T-F	2.99	-	4.21	3.55	3.62	-	6.12
Self-Adapt MHSA, [67]	T-F	2.99	-	4.15	3.42	3.57	-	-
Low-latency U-Net, [68]	T-F	2.90	94	4.22	3.32	3.58	-	-
NAAGN, [69]	T-F	2.90	-	4.13	3.50	3.51	-	-
DEMUCS, [27]	T	**3.07**	**95**	4.31	3.4	3.63	33.5	17.27
T-GSA, [49]	T-F	3.06	-	4.18	3.59	3.62	63.0	-
CARN, [71]	T-F	2.93	**95**	4.19	3.61	3.54	-	-
GaGNet [76]	T-F	2.94	-	4.26	3.45	3.59	5.94	1.63
FSCNet [77]	T-F	3.05	-	4.28	3.57	3.68	0.9	
SADNUNet [78]	T	2.82	-	4.18	3.47	3.51	2.63	-
TSTNN [79]	T	2.96	**95**	4.33	3.53	3.67	0.92	75.52
CPTNN [80]	T	**3.07**	**95**	**4.40**	3.59	**3.76**	0.76	29.83
SE-Mixer (ours)	T-F	3.05	**95**	4.20	3.50	3.63	**0.71**	0.21

slightly inferior to T-GSA (3.06) using transformer with gaussian-weighted self-attention and CPTNN (3.07) using cross-parallel transformer. TSTNN using two-stage transformer performs slightly better in MOSs metrics in terms of CSIG and COVL, while has about 0.1 reduction of PESQ compared with SE-Mixer. Additionally, SE-Mixer has the fewest trainable parameters which is about 1.3 times fewer than TSTNN and approximately 155 times fewer than T-GSA, respectively. From the above observations, we found that the proposed SE-Mixer still has a superior performance in speech enhancement even though it is free of attention mechanism. Moreover, SE-Mixer has lowest multiply-accumulate operation (MACs) which is approximately 345 times fewer than TSTNN and about 141 times fewer than CPTNN, indicating an extremely

low computation complexity.

3.3.3 Ablation study

In this section, we further explore the performance of our proposed SE-Mixer with different network configurations. First, the proposed mixer block, as an important component, is designed to effectively extract the temporal and frequency features through the proposed temporal MLP and frequency MLP. In order to explore the impact of the number of mixer blocks on speech enhancement, we here set up two comparison models including 4 and 6 mixer blocks as shown in Table 9, namely SE-Mixer_Small and SE-Mixer_Mid, respectively. Moreover, for a fair comparison, we maintain other configurations consistent, including time scales in temporal MLP.

Table 9: Network configuration with different blocks and time scales

	Time_Scales	**Blocks**	**Param.**
SE-Mixer	4	8	710k
SE-Mixer_Mid	4	6	549k
SE-Mixer_Small	4	4	388k
SE-Mixer_FixedScale	4 & fixed	8	710k
SE-Mixer_2Scales	2	8	426k
SE-Mixer_1Scale	1	8	284k

Additionally, our proposed multi-scale convolution block is applied to extract temporal features from different time scales by controlling the dilation rate in each bottleneck dilated convolution, which plays a similar role as multi-head attention in transformer. Different from transformer, our proposed multi-scale convolution block is free of attention mechanism and resorts to basic matrix multiplication and changes of feature layout (reshapes). Therefore, we further analyze the influence of time scales in temporal Mixer on performance by designing another three comparison models as indicated in Table 9. We first use the same number of bottleneck dilated convolutions in multi-scale convolution block, where the dilation rates are all set as 1 (equal to 2D-convolution with kernel size of (1, 3)), leading to the same time scales in temporal MLP. Next, we set the number of bottleneck dilated convolution as 1 and 2 respectively, leading to the other two comparison models with 1 and 2 time scales in temporal MLP.

Table 10: Experimental results of different configurations

	PESQ	STOI	CSIG	CBAK	COVL
SE-Mixer	**3.05**	**95%**	4.20	**3.50**	3.63
SE-Mixer_Mid	3.04	94%	**4.22**	3.45	**3.64**
SE-Mixer_Small	2.99	94%	4.15	3.44	3.58
SE-Mixer_FixedScale	2.98	94%	4.15	3.45	3.57
SE-Mixer_2Scales	2.97	94%	4.16	3.44	3.58
SE-Mixer_1Scale	2.96	94%	4.10	3.42	3.53

As presented in Table 10, the performance scores of speech enhancement steadily grow as the number of mixer blocks increases, especially PESQ and STOI. It is worth mentioning that SE-Mixer_Small still reaches the remarkable results in PESQ (2.99) as well as other evaluation metrics with only 384k trainable parameters, indicating that the proposed mixer block is highly efficient for dealing with long-range speech sequences. Moreover, in contrast to the comparison models with different time scales, SE-Mixer outperforms other configurations using fixed or fewer time scales in all objective metrics, revealing that adopting enough and different time scales in our temporal Conv-MLP is notably crucial for extracting sufficient temporal features and advancing the performance of speech denoising.

Table 11: Experimental results of different denoising blocks

	Time Mixing	Frequency Mixing	PESQ	STOI	CSIG	CBAK	COVL
SE-Mixer	√	√	**3.05**	**95%**	**4.20**	**3.50**	**3.63**
SE-Temporal	√	×	3.00	94%	4.16	3.48	3.59
SE-Frequency	×	√	2.65	94%	3.94	3.27	3.29
SE-Transformer	×	×	2.60	93%	3.42	3.11	2.99

Finally, to analyze the components of the proposed mixer block, we design two comparison models by removing temporal Conv-MLP or frequency MLP, yielding SE-Frequency and SE-Temporal as shown in Table III. To compare the proposed SE-Mixer with transformer using the same architecture and input features, we replace the mixer block by several improved transformer proposed from TSTNN, leading to the SE-Transformer comparison model. As

indicated in Table 11, omitting temporal Conv-MLP or frequency MLP performs worse than the SE-Mixer retaining temporal MLP and frequency MLP, presenting that temporal mixing and frequency mixing are crucial for processing long-range speech sequence. Especially, SE-Frequency has obviously inferior evaluation scores than SE-Mixer, in terms of PESQ and MOS scores, showing that the temporal Conv-MLP with multi-scale operation plays an important role in speech enhancement. Moreover, compared with SE-Transformer, our SE-Mixer achieves impressive improvement on evaluation scores, revealing that our proposed attention-free mixer block is very efficient for processing long-range speech sequences.

3.4 Summary

In this chapter, we have proposed an attention-free architecture as an alternative to the transformer for long-range sequence modeling in speech enhancement. This new structure, named as SE-Mixer, consists of an encoder, multiple mixer blocks and a decoder, all based on the MLP, matrix multiplication and changes of data layout. The encoder encodes the speech spectrogram to obtain the speech features via a FC layer. The mixer blocks are proposed to extract contextual information of speech feature, where each block is comprised of a temporal MLP and frequency MLP. The temporal MLP mixes the temporal information among different time scales by introducing the multi-scale convolution into MLP, where the dilated convolution used in multi-scale convolution controls the time scales. The frequency MLP mixes the frequency information by adopting a simple MLP. By successively extracting temporal and frequency information, the contextual information is obtained and further processed by a FC layer to create a mask, which will be multiplied with input magnitude spectrogram to get the enhanced magnitude spectrogram. Finally, the noisy phase is used along with the enhanced magnitude for reconstructing the enhanced speech waveform via iSTFT.

The experimental results show that our proposed SE-Mixer achieves a competitive performance among existing methods including attention- or transformer-based models while involving lowest model complexity. We have also shown that the architecture without attention algorithm is likely to reach approximately the same effectiveness as compared with attention-assisted models but with significantly lower computational complexity.

Chapter 4
Conclusions and Future work

4.1 Summary of the work

In this thesis, novel deep-learning-based approaches have been proposed for single-channel speech enhancement. More specifically, the efficient neural networks including attention-based transformer neural networks and attention-free multi-layer perceptrons have been investigated for learning the long-term dependency in speech enhancement. In addition, designing a speech enhancement system with a low footprint and computational complexity is the other important goal of this thesis.

In Chapter 2, two neural network models based on transformer are proposed for sequence modeling in speech enhancement, where the transformer has been widely studied in NLP tasks and shows superiority in extracting long-term dependency of long sequences. We first propose a two-stage transformer neural network (TSTNN) to extract local and global information by using local and global transformer with cascaded connection, respectively. Specifically, the local transformer is implemented on the individual frame to learn the short-term relationship. Then, the global transformer is used to summarize the features across the various frames to capture the long-term information. After iteratively learning the local and global information, the contextual feature representation is obtained for speech denoising. To further resolve the information leakage existing in cascaded structure, we propose an improved version of transformer with cross-parallel structure, named CPTNN, to parallel perform the local and global extraction. The cross-attention based transformer is employed to dynamically fuse the outputs of two-branch extraction, generating the efficient contextual information of speech features. Proposed TSTNN and CPTNN share a similar structure including an encoder, a masking module and a decoder, except for the intermediate transformer blocks. The outputs from transformer blocks are passed through the masking module to create a mask, which will be multiplied with encoder outputs to remove the noise-related information. Through the conducted experiments, we have shown that our proposed TSTNN and CPTNN obtain the competitive performance compared with existing

state-of-the-art methods while involving considerably fewer trainable parameters. Especially for CPTNN, it achieves a superior performance than other current methods in the most of evaluation metrics.

In Chapter 3, we propose an attention-free architecture, named SE-Mixer, to verify whether attention- or transformer-based networks are indispensable for resolving the long-term dependency problem in speech enhancement. It consists of an encoder, a decoder and stacked mixer blocks in between. The encoder is used to transform the input speech into compressed speech features which will be processed by the mixer blocks for capturing the contextual information. Each mixer block is comprised of temporal MLP and frequency MLP to perform the temporal mixing and frequency mixing, respectively, where the temporal MLP extracts the multi-scale temporal information from different time steps, and the frequency MLP captures the abundant frequency information at individual time step. The decoder is adopted to convert the contextual speech features into the enhanced speech by masking method. Our experimental results reveal that the proposed SE-Mixer achieves a comparable performance to existing models with and without attention mechanism while having the fewest parameters and complexity. We also show that the attention-free models have capacity to attain the same or even better effectiveness compared with attention-augmented models but with significantly lower computational complexity.

4.2 Future work

This thesis mainly focuses on the deep learning approaches for single-channel speech enhancement. However, many real-word applications like microphone, involve the multi-channel speech enhancement system. Fortunately, our proposed neural networks are relatively flexible and could be implemented on multi-channel speech enhancement with a few modifications. In addition, our proposed models are non-casual systems for speech enhancement, which means that the denoising begins after looking through the whole utterance. Many real-life applications require speech enhancement systems to have real-time property, such as real-time online meeting scenario. It would be interesting to see the casual or real-time performance of the proposed models with casual setting.

Furthermore, our proposed neural network models are trained and evaluated on the public

VoiceBank-Demand dataset. Some studies have revealed that DNN-based methods fail to generalize to untrained corpora even though they perform well on a trained corpus [81]. We plan to train our models on the larger datasets such as LibriSpeech and then evaluate it on VoiceBank-Demand to explore the cross-corpus generalization of the transformer-based and attention-free architectures.

Bibliography

[1] P. Scalart et al., "Speech enhancement based on a priori signal to noise estimation," in 1996 *IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, vol. 2. IEEE, 1996, pp. 629–632.

[2] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech, and Signal processing*, vol. 27, no. 2, pp. 113–120, 1979.

[3] P. C. Loizou, *Speech Enhancement: Theory and Practice*, 2nd ed. Boca Raton, FL, USA: CRC Press, 2013.

[4] N. Saleem and M. I. Khattak, "Deep neural networks for speech enhancement in complex-noisy environments," *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 6, no. 1, pp. 84–90, 2020.

[5] U. Kjems, J. B. Boldt, M. S. Pedersen, T. Lunner, and D. Wang, "Role of mask pattern in intelligibility of ideal binary-masked noisy speech," The *Journal of the Acoustical Society of America*, vol. 126, no. 3, pp. 1415–1426, 2009.

[6] C. Hummersone, T. Stokes, and T. Brookes, "On the ideal ratio mask as the goal of computational auditory scene analysis," in *Blind source separation*. *Springer*, 2014, pp. 349–368.

[7] D. S. Williamson, Y. Wang, and D. Wang, "Complex ratio masking for monaural speech separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 3, pp. 483–492, 2015.

[8] K. Paliwal, K. W´ojcicki, and B. Shannon, "The importance of phase in speech enhancement," *Speech Communication*, vol. 53, no. 4, pp. 465–494, 2011.

[9] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *ICML*, 2010.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[11] M. Shahid, "Convolutional neural network," *https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529*.

[12] S.-W. Fu, T.-y. Hu, Y. Tsao, and X. Lu, "Complex spectrogram enhancement by convolutional neural network with multi-metrics learning," in 2017 *IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2017, pp. 1–6.

[13] S. R. Park and J. Lee, “A fully convolutional neural network for speech enhancement,” *arXiv preprint arXiv:1609.07132*, 2016.

[14] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] K. Cho, B. Van Merri ̈enboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches*,” arXiv preprint arXiv:1409.1259*, 2014.

[16] H. Zhao, S. Zarar, I. Tashev, and C.-H. Lee, “Convolutional-recurrent neural networks for speech enhancement,” in 2018 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 2401–2405.

[17] K. Tan, and D. Wang. "A Convolutional Recurrent Neural Network for Real-Time Speech Enhancement." In *INTERSPEECH*, 2018, pp. 3229-3233.

[18] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in *International Conference on Machine Learning*. PMLR, 2015, pp. 448–456.

[19] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, “Fast and accurate deep network learning by exponential linear units (elus*),” arXiv preprint arXiv:1511.07289*, 2015.

[20] Y. Hu, Y. Liu, S. Lv, M. Xing, S. Zhang, Y. Fu, J. Wu, B. Zhang, and L. Xie, “Dccrn: Deep complex convolution recurrent network for phase- aware speech enhancement,” *arXiv preprint arXiv:2008.00264*, 2020.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Delving deep into rectifiers: Surpassing human-level performance on imagenet classification,” in *IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[22] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 2015, pp. 234–241.

[23] C. Macartney and T. Weyde, “Improved speech enhancement with the wave-u-net*,” arXiv preprint arXiv:1811.11307*, 2018.

[24] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” *arXiv preprint arXiv:1511.07122*, 2015.

[25] K. Tan, J. Chen, and D. Wang, “Gated residual networks with dilated convolutions for monaural speech enhancement,” *IEEE/ACM transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 189–198, 2018.

[26] Pandey, A., & Wang, D. "Densely Connected Neural Network with Dilated Convolutions for Real-Time Speech Enhancement in The Time Domain." In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6629-6633.

[27] Defossez A, Synnaeve G, Adi Y, "Real Time Speech Enhancement in the Waveform Domain". *arXiv preprint arXiv:2006.12847*, 2020.

[28] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," *in International Conference on Machine Learning*. PMLR, 2017, pp. 933–941.

[29] . Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[30] V. Kishore, N. Tiwari, and P. Paramasivam, "Improved speech enhancement using tcn with multiple encoder-decoder layers." in *Interspeech*, 2020, pp. 4531–4535.

[31] A. Pandey and D. Wang, "TCNN: Temporal convolutional neural network for real-time speech enhancement in the time domain," in *ICASSP*, 2019, pp. 6875–6879.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[33] Z. Kong, W. Ping, A. Dantrey, and B. Catanzaro, "Speech denoising in the waveform domain with self-attention," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7867–7871.

[34] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon technical report* n, vol. 93, p. 27403, 1993.

[35] Valentini-Botinhao C, Wang X, Takaki S, et al. "Investigating RNN-based speech enhancement methods for noise-robust Text-to-Speech." In *SSW*. 2016: 146-152.

[36] Veaux C, Yamagishi J, King S. "The voice bank corpus: Design, collection and data analysis of a large regional accent speech database." In 2013 *International Conference Oriental COCOSDA* held jointly with 2013 *Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*. IEEE, 2013: 1-4.

[37] J. Thiemann, N. Ito, and E. Vincent, "The diverse environments multi-channel acoustic noise database (DEMAND): A database of multichannel environmental noise recordings," in *Proceedings of Meetings on Acoustics ICA2013*, vol. 19, no. 1. Acoustical Society of America, 2013, p. 035081.

[38] C. K. Reddy, V. Gopal, R. Cutler, E. Beyrami, R. Cheng, H. Dubey, S. Matusevych, R. Aichner, A. Aazami, S. Braun et al., "The interspeech 2020 deep noise suppression challenge: Datasets, subjective testing framework, and challenge results," *arXiv preprint arXiv:2005.13981*, 2020.

[39] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[40] C. H. Taal et al., "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[41] J. H. Hansen and B. L. Pellom, "An effective quality evaluation protocol for speech enhancement algorithms." in *ICSLP*, vol. 7. Citeseer, 1998, pp. 2819–2822.

[42] Hu Y, Loizou P C. "Evaluation of objective quality measures for speech enhancement." *Audio, Speech, and Language Processing*, IEEE Transactions on, 2008, 16(1): 229-238.

[43] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[44] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv pre-print arXiv:1607.06450*, 2016.

[45] Y. Wu and K. He, "Group normalization," in *Proceedings of the Euro-pean conference on computer vision (ECCV)*, 2018, pp. 3–19.

[46] B. Yin, H. S. Scholte, and S. Bohtˊe, "Localnorm: Robust image classification through dynamically regularized normalization," in *International Conference on Artificial Neural Networks*. Springer, 2021, pp. 240–252.

[47] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[49] J. Kim, M. El-Khamy, and J. Lee, "T-GSA: Transformer with gaussian-weighted self-attention for speech enhancement," *in ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6649–6653.

[50] E. Kim and H. Seo, "SE-Conformer: Time-Domain Speech Enhancement Using Conformer," in *INTERSPEECH*, 2020, pp. 2736–2740.

[51] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-path RNN: efficient long se-quence modeling for time-domain single-channel speech separation," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 46–50.

[52] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 21–25.

[53] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," *arXiv preprint arXiv:1910.05895*, 2019.

[54] Y. Wang, A. Mohamed, D. Le, C. Liu, A. Xiao, J. Mahadeokar, H. Huang, A. Tjandra, X. Zhang, F. Zhang, et al., "Transformer-based acoustic modeling for hybrid speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6874–6878.

[55] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[56] F. Chollet, "Xception: Deep learning with depth-wise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1251–1258.

[57] W. Shi, J. Caballero, F. Huszar, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1874–1883.

[58] A. Pandey and D. Wang, "Dense CNN with self-attention for time-domain speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1270–1279, 2021.

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, vol. 9, 2015.

[60] J Chen, Q Mao, D. Liu "Dual-Path Transformer Network: Direct Context-Aware Modeling for End-to-End Monaural Speech Separation". *arXiv preprint arXiv:2007.13975*, 2020.

[61] S. Pascual, A. Bonafonte, and J. Serra, "SEGAN: Speech enhancement generative adversarial network," *arXiv preprint arXiv:1703.09452*, 2017.

[62] D. Rethage, J. Pons, and X. Serra, "A wavenet for Speech Denoising," in 2018 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5069–5073.

[63] R Giri, U Isik, and A Krishnaswamy, “Attention wave-u-net for speech enhancement,” in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 249–253.

[64] S.-W. Fu, C.-F. Liao, Y. Tsao, and S.-D. Lin, “MetricGAN: Generative adversarial networks based black-box metric scores optimization for speech enhancement,” in *International Conference on Machine Learning*. PMLR, 2019, pp. 2031–2041.

[65] H.-S. Choi, J.-H. Kim, J. Huh, A. Kim, J.-W. Ha, and K. Lee, "Phase-aware speech enhancement with deep complex u-net." in *International Conference on Learning Representations*. 2018.

[66] D. Yin, C. Luo, Z. Xiong, and W. Zeng, “PHASEN: A phase-and-harmonics-aware speech enhancement network,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 9458–9465.

[67] Y. Koizumi, K. Yatabe, M. Delcroix, Y. Masuyama, and D. Takeuchi, “Speech enhancement using self-adaptation and multi-head self-attention,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 181–185.

[68] A. E. Bulut and K. Koishida, “Low-latency single channel speech enhancement using u-net convolutional neural net-works,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6214–6218.

[69] F. Deng, T. Jiang, X. Wang, C. Zhang, and Y. Li, “NAAGN: Noise-aware attention-gated network for speech enhancement.” in *INTERSPEECH*, 2020, pp. 2457–2461.

[70] K. Wang, B. He, and W.-P. Zhu, “CAUNet: Context-aware u-net for speech enhancement in time domain,” in 2021 *IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2021, pp. 1–5.

[71] L. Zhou, Y. Gao, Z. Wang, J. Li, and W. Zhang, “Complex spectral mapping with attention-based convolution recurrent neural network for speech enhancement,” *arXiv pre-print arXiv:2104.05267*, 2021.

[72] J. Lin, A. J. van Wijngaarden, K.-C. Wang, and M. C. Smith, “Speech enhancement using multi-stage self-attentive temporal convolutional networks,” *arXiv preprint arXiv:2102.12078*, 2021.

[73] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, D. Keysers, J. Uszkoreit, M. Lucicet al., “Mlp-mixer: An all-mlp architecture for vision,” *arXiv preprint arXiv:2105.01601*, 2021.

[74] O. Tatanov, S. Beliaev, and B. Ginsburg, “Mixer-tts: non-autoregressive, fast and compact text-to-speech model conditioned on language model embeddings,” in *ICASSP 2022-2022*

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2022, pp. 7482– 7486.

[75] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *arXiv preprint arXiv:1608.03983*, 2016.

[76] A. Li, C. Zheng, L. Zhang, and X. Li, "Glance and gaze: A collaborative learning framework for single-channel speech enhancement," *Applied Acoustics*, vol. 187, p. 108499, 2022.

[77] L. Cheng, J. Li, and Y. Yan, "Fscnet: Feature-specific convolution neural network for real-time speech enhancement," *IEEE Signal Processing Letters*, vol. 28, pp. 1958–1962, 2021.

[78] X. Xiang, Z. Xiaojuan, and C. Haozhe, "A nested u-net with self-attention and dense connectivity for monaural speech enhancement," *IEEE Signal Processing Letters*, 2021.

[79] K. Wang, B. He, and W.-P. Zhu, "TSTNN: Two-stage transformer based neural network for speech enhancement in the time domain," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 7098–7102.

[80] K. Wang, B. He, and W.-P. Zhu, "CPTNN: Cross-Parallel transformer based neural network for time-domain speech enhancement," accepted by *IWAENC 2022 IEEE International Workshop on Acoustics Signal Enhancement (IWAENC)*.

[81] A. Pandey and D. L. Wang, "On cross-corpus generalization of deep learning based speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2489–2499, 2020.