2024 27th Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA) | 979-8-3315-0603-2/24/$31.00 ©2024 IEEE | DOI: 10.1109/O-COCOSDA64382.2024.10800743

IMPROVING REAL-TIME MUSIC ACCOMPANIMENT SEPARATION WITH MMDENSENET

Chun-Hsiang Wang[1], *Chung-Che Wang*[1], *Jun-You Wang*[2], *Jyh-Shing Roger Jang*[1], *Yen-Hsun Chu*[3]

[1]Dept. of CSIE, National Taiwan University, Taiwan
[2]Institute of Information Science, Academia Sinica, Taiwan
[3]Realtek Semiconductor Corp.

ABSTRACT

Music source separation aims to separate polyphonic music into different types of sources. Most existing methods focus on enhancing the quality of separated results by using a larger model structure, rendering them unsuitable for deployment on edge devices. Moreover, these methods may produce low-quality output when the input duration is short, making them impractical for real-time applications. This challenge is akin to those in speech processing models and systems, where isolating and analyzing specific audio components is critical. Therefore, the goal of this paper is to enhance a lightweight model, MMDenstNet, to strike a balance between separation quality and latency for real-time applications. Different directions of improvement are explored or proposed in this paper, including complex ideal ratio mask, self-attention, band-merge-split method, and feature look back. Source-to-distortion ratio, real-time factor, and optimal latency are employed to evaluate the performance. To align with our application requirements, the evaluation process in this paper focuses on the separation performance of the accompaniment part. Experimental results demonstrate that our improvements achieve a low real-time factor and optimal latency while maintaining a comparable source-to-distortion ratio.

***Index Terms*—** MMDenseNet, complex ideal ratio mask, self-attention, band-split method, feature look back

1. INTRODUCTION

Music source separation aims to separate polyphonic music into different types of sources, such as vocals, drums, piano, or other instruments. Voice separation is not only a research topic in itself, but achieving good separation results can also benefit downstream music or speech-related applications, including voice conversion [1], lyrics alignment [2], and accompaniment for karaoke [3]. Moreover, the techniques used in music source separation are closely related to those in speech processing models and systems, where isolating specific audio components is essential. In this paper, we aim to extract the accompaniment part in real-time with low latency for a karaoke application.

With the rapid development of deep learning, the effectiveness of singing voice separation has significantly improved. HT Demucs [4] is the state-of-the-art model (among those with official implementation before mid-2024), which employs two U-Nets that respectively process waveform and spectrogram. It combines the information from both U-Nets using a Transformer and cross-attention at the bottleneck layer. Despite the high separation quality of HT Demucs, its larger number of parameters and high latency on CPU make it challenging for real-time applications on edge devices. On the other hand, the Multi-scale multi-band DenseNet (MMDenseNet) [5] is one of the relatively lightweight source separation models. Despite MMDenseNet's slightly lower separation quality, its fast speed makes it more suitable for real-time applications.

To further reduce the latency while still maintaining comparable separation quality, we propose or invoke various methods for improving the raw version of MMDenseNet, including the complex ideal ratio mask (cIRM) [6], self-attention, the band-merge-split method inspired by [7], and feature look back. Note that the above methods are possible for benefiting other separation model structures, we choose MMDenseNet for this paper because our preliminary experiments show that MMDenseNet achieves the best trade-off for both quality and speed for our application requirements.

The rest of this paper is organized as follows. Section 2 describes our methods for improving MMDenstNet, Section 3 shows the experimental results, and Section 4 concludes this paper and addresses possible future work.

2. METHODS

In this Section, brief description of MMDenseNet [5] is first given, followed by the four invoked or proposed methods, complex ideal ratio mask (cIRM) [6], self-attention, the band-merge-split method inspired by [7], and feature look back, for improving MMDenseNet. While the first three methods aim to improve the separated audio quality, the goal of the feature look back method is to reduce the latency with little degradation of the separated audio quality.

Thanks to the National Center for High-performance Computing (NCHC) for providing computational and storage resources.

2.1. The Original MMDenseNet

MMDenseNet [5] was proposed by Takahashi and Mitsufuji, which won the signal separation evaluation campaign in 2016 [8] with less number of parameters and faster training speed than other models in the evaluation campaign. The basic component of MMDenseNet is DenseNet [9] or dense block, which is mainly used to comprise the multi-scale DenseNet (MDenseNet). The MDenseNet is a U-Net like structure, where the encoder part is composed of DenseNets and down-sampling layers, and the decoder part is composed of DenseNets and up-sampling layers. The input spectrogram is splitted into N subbands, and N MDenseNets are used for processing the N subbands, and 1 MDenseNet is used for processing the full band (i.e. all frequency bands). Finally, 1 DenseNet is used to combine the outputs of the $N + 1$ MDenseNets for obtaining the spectrogram of the separated signal. For the structure of MMDenseNet, please refer to the original paper [5].

2.2. Complex Ideal Ratio Mask

While the final results of a source separation task should be spectrograms or signals, the raw output of the neural network in this task can be in different forms, and post-processing techniques can be applied for the output of the network. One of the popular methods is to use the magnitude mask as the output of the separation model, and the output mask is then element-wise multiplied with the input spectrogram to obtain the final output. Besides, for some source separation models, Wiener filter [10] is used for post-processing to obtain a better final separated results. However, since Wiener filter utilizes all the separated sources for computation, requires a larger amount of computational resources and is unsuitable for this paper (details are shown in Section 3.3).

In this paper, we investigate the performance of using both magnitude and phase estimation, where their effectiveness has been shown in [6], as the new output form for MMDenseNet. The modified network structure is shown in Fig. 1, where $\hat{M}_{mag}$, $\hat{Q}$, $\hat{P}_r$ and $\hat{P}_i$ are respectively magnitude mask estimation, magnitude estimation, phase estimation of real part, and phase estimation of imaginary part. $\hat{M}_{mag}$ and $\hat{Q}$ are used to estimate the magnitude spectrogram, $\hat{P}_r$ and $\hat{P}_i$ are used to estimate the phase spectrogram, and the final separated result is obtained using estimated magnitude and phase spectrogram and inverse short-time Fourier transform. F and N in Fig. 1 are respectively 1,025 and 2, and T varies in different experiments.

2.3. Self-Attention

Due to the great success of using self-attention (SA) in various scenarios [3], we propose adjusted self-attention structures and apply it to MMDenseNet. The adjusted structure of the self-attention along the time axis is shown in Fig. 2,

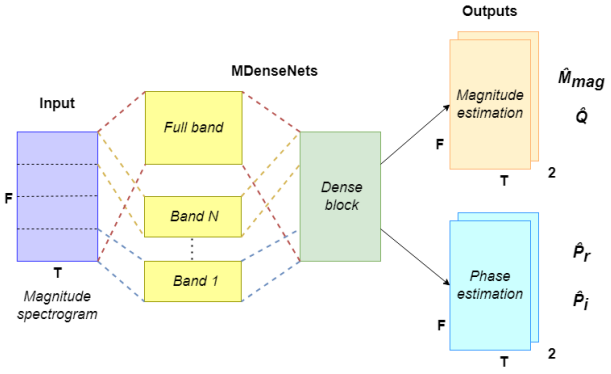

Fig. 1. The modified MMDenseNet which uses cIRM as the new output form.

where different channels are viewed as different attention heads, and pointwise (PW) convolution layers are used to reduce the computational cost by decreasing the number of channels. The adjusted structure of the self-attention along frequency axis is similar to that of the self-attention along time axis, but chunkwise self-attention is applied, and the residual connection between module input and output is not used. In our setting for chunkwise self-attention, the input spectrogram with length T of the time axis is split into T/t chunks, where t is set to 16 in this paper. Note that our attention structures are similar to that of TF-GridNet [11], with the following differences: 1) we use layer normalization, 2) normalization is applied only at the beginning of the attention structures, and 3) PReLU is not used.

To apply self attention to the MMDenseNet, we attach the self-attention modules after each dense block of a fullband or subband MDenseNet except the first block. The self-attention along time axis is used for both fullband and subband MDenseNets, but the self-attention along frequency axis is only used for fullband MDenseNet. When using both the self-attention along time and frequency axis for fullband MDenseNet, the self-attention along frequency axis is used before the self-attention along time axis.

2.4. Band-Merge-Split Method

The structures of MMDenseNet and band-split RNN [7] (BSRNN) are similar in some way since both of them split the input spectrogram into subbands. However, a module similar to the band and sequence modeling module of BSRNN which captures information across time and frequency bands does not exists in MMDenseNet. Therefore, inspired by BSRNN, we use the band-merge-split method for MMDenseNet, and the modified structure is shown in Fig. 3, where DS and US are respectively Down Sampling and Up Sampling.

The band-merge-split method connects two subband MDenseNets and includes three modules: band-merge, cross-band attention, and band-split. The band-merge module concatenates features from the middle of different MDenseNets

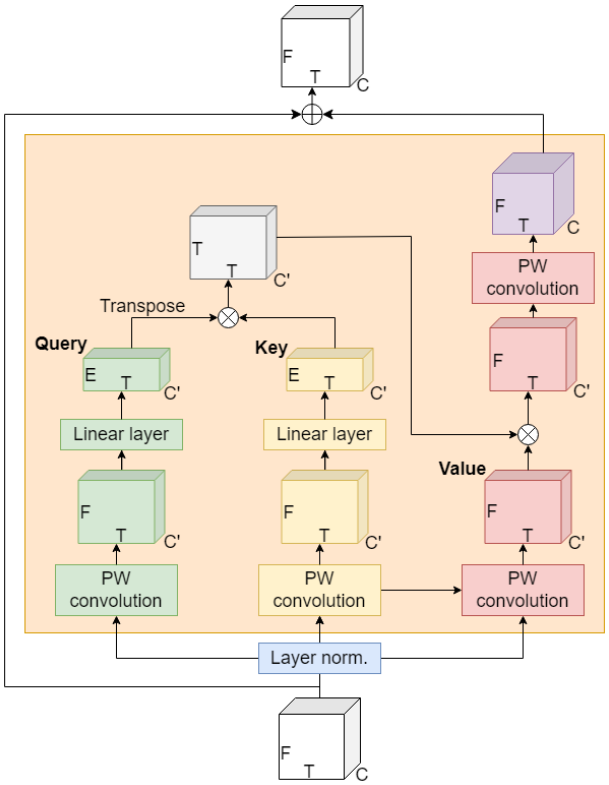

Fig. 2. The adjusted structure of the self-attention along time axis. E and C' are respectively set to 20 and 5 in this paper.

along the frequency axis. Due to the fact that the features from different MDenseNets have different number of channels, pointwise convolution [12] is applied to adjust the number of channels in one of the features before concatenation. The cross-band attention module is used to share information across frequency bands. In order to let each feature be concentrated on its corresponding important frequency bands and time periods, self attention is applied to both the frequency and time axes. The band-split module splits the feature along the frequency axis, and pointwise convolution is applied again to adjust the number of channels back to its original number for further processing.

2.5. Feature Look Back

Intuitively, source separation models favors longer input and produce low quality separation results when the input duration is short. To maintain the separation quality when the input duration is short, feature look back (FLB) is used to combine past and current information for obtaining the separation output for the current input. To precisely describe feature look back, following terms should be defined:

- Training segment: model input at the training stage, consists of one or multiple training chunks. The size of a training segment is denoted by $N_{\text{tr-s}}$ frames.
- Training chunk: model input (consists of multiple frames) at each time at the training stage. The size of a training chunk is denoted by $N_{\text{tr-c}}$ frames. Note that $N_{\text{tr-c}}$ should divide $N_{\text{tr-s}}$.
- Training look back chunk: corresponding model input of look back information at each time at the training stage. The size of a training look back chunk is denoted by $N_{\text{tr-lbc}}$ frames.
- Test chunk: model input at each time at the test stage. The size of a test segment is denoted by $N_{\text{te-s}}$ frames.
- Test look back chunk: corresponding model input of look back information at each time at the test stage. The size of a test look back chunk is denoted by $N_{\text{te-lbc}}$ frames.

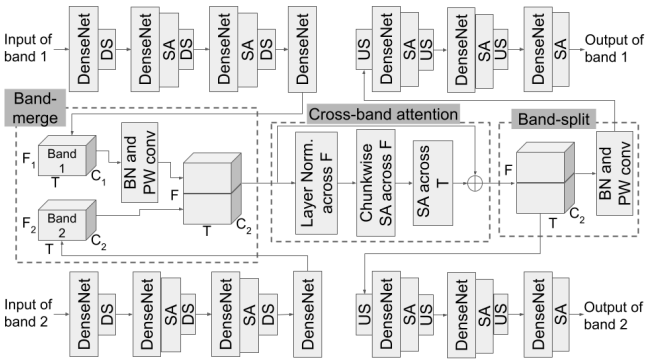

Fig. 3. Our band-merge-split method and its connections to the two subband MDenseNets. Skip connections between DenstNets are omitted in this figure.

The latency is reduced by shorten the model input from $N_{\text{tr-s}}$ frames to $N_{\text{tr-c}}$ frames at each time, and the separation quality is possible to be maintained by looking back of past features.

At the training stage, a training segment is split into training chunks, the chunks are feed into the network one by one, and the loss is calculated once all the chunks of a segment are processed by the network. If feature look back is enabled, the hidden representation of the past $N_{\text{tr-lbc}}$ frames are considered together with a current training chunk when processing a current training chunk. The process of the test stage when feature look back is enabled is similar. Test chunks are feed into the network one by one, and past $N_{\text{te-lbc}}$ frames are considered together with a current test chunk when processing a current test chunk.

The network structure for each MDenseNet when feature look back is enabled is shown in Fig. 4, where A, B, and C are different possible look back connections. For simplicity, we only show the case when $N_{\text{tr-c}}$ or $N_{\text{te-c}}$ is equal to $N_{\text{tr-lbc}}$ or $N_{\text{te-lbc}}$ (i.e. look back only for one previous training or test chunk). In the experimental Section, we show the results of using only connection A, using both connections A and B (denoted as A+B), and using connections A, B, and C (denoted as A+B+C). Note that our feature look back is different from memory transformers [13,14] since the output size is not

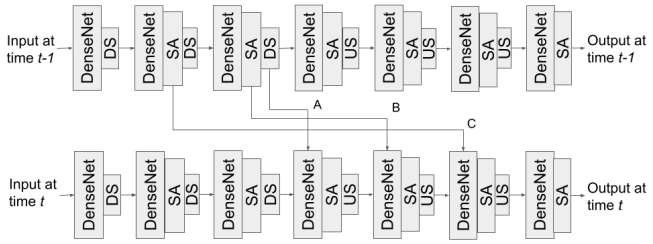

Fig. 4. Network structure for each MDenseNet when feature look back is enabled. The input at one timestamp is a training or test chunk. Skip connections between DenstNets are omitted in this figure. A, B, and C are different possible look back connections.

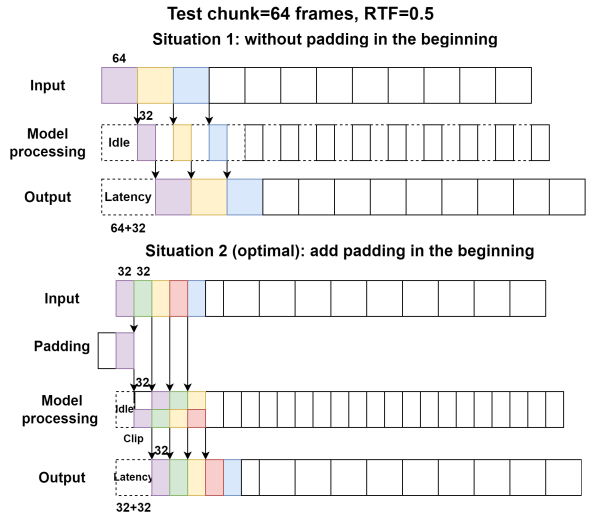

Fig. 5. The relationship between the duration of a test chunk, RTF, and latency. Different colors indicate test chunks at different timestamps.

changed (because the output of the SA blocks after DenseNets are cropped).

3. EXPERIMENTAL RESULTS

3.1. Evaluation Metrics

Source-to-distortion ratio (SDR) [15], real-time factor (RTF), and optimal latency are used to evaluate the model performance. SDR is used as the primary evaluation metric in many previous studies [4,5]. For a song track, the SDR for each one second segment is first calculated using Museval [16], and the song track's SDR is calculated as the median of all one-second segments' SDR. A higher SDR indicates the better separation results. The RTF is defined as the total processing time of a song track divided by the total duration of the song track, and the optimal latency could be therefore calculated as the RTF multiply by the duration of a test chunk multiply by two. The lower RTF and optimal latency indicate that the model runs faster.

The derivation from RTF to optimal latency is shown as follows. Since the case of $RTF \geq 1$ does not meet our application requirement, we consider only the case of $RTF \leq 1$. Generally, latency is the duration of a test chunk (T) plus the model processing time ($T + RTF \times T$), as illustrated in the upper part of Fig. 5. However, by padding a blank signal of duration t at the beginning and feeding a signal of duration T into the model every $RTF \times T$ time units (where $t \geq RTF \times T$), as depicted in the lower part of Fig. 5, the latency can be calculated as follows, and the latency is optimal when the equality holds:

$$\begin{aligned} Optimal\ Latency &= t + RTF \times T \\ &\geq RTF \times T + RTF \times T \\ &= 2 \times RTF \times T \end{aligned} \quad (1)$$

3.2. Dataset and Experimental Setup

MUSDB18 [17] is used in this study. It contains 150 music tracks of different genres with 44.1 kHz sampling rate. Each track is composed of four channels, including vocal, drum, bass, and the rest of the accompaniment. The goal in this study, accompaniment separation, is to separate the mixture of drum, bass, and the rest of the accompaniment. 86 and 14 tracks are respectively used for training and validation, and 50 tracks are used as the test set for calculating SDR. For RTF calculation, a set of 8 privately collected tracks with an average duration of about 243 seconds are used.

Experiments are conducted in two different machines. The first one is a container running on a machine with Intel(R) Xeon(R) Gold 6154 CPU, occupying 4 CPU cores, 90 GB of host memory, and a Tesla V100-SXM2 GPU. The second one has an Intel(R) Xeon(R) Silver 4116 CPU, 328 GB of RAM, an NVIDIA Quadro GV100, and an NVIDIA TITAN RTX GPU. For training and test, we use one of the machines simply based on their availability at that time. But for RTF calculation, we run our implementation on the CPU of the later one, and the number of used CPU cores is limited to 1.

The window size and the hop size for short-time Fourier transform are respectively $2,048$ and $1,024$, and the Hann function is used for windowing. The size of the training segment, training chunk, training lookback chunk, test chunk, and test lookback chunk varies in different experiments due to application requirements and will be described in the following subsection. The L1 loss and the ADAM optimizer are used. The initial learning rate is 10^{-3} and is multiplied by 0.99 every 5 epochs. Due to resource limitations, the batch size is set to 8 or 16 for different experiments, and we empirically stop the training process when the validation loss is not significantly decreasing, which is usually no more than 500 epochs. All models are trained from scratch.

Config Name	cIRM	SA	BS	Num of params (k)	Training			Test		SDR	RTF	Optimal latency (s)
					Seg	Chunk	LBC	Chunk	LBC			
Raw MMDenseNet	No	No	No	339	256	256	No	256	No	11.162	0.368	4.38
Raw MMDenseNet (with Wiener filter)	No	No	No	679	256	256	No	256	No	13.872	0.841	9.99
Raw MMDenseNet (Mag. mask as output)	No	No	No	339	256	256	No	256	No	13.555	0.394	4.68
cIRM	Yes	No	No	343	256	256	No	256	No	13.951	0.370	4.39
cIRM+SA_T	Yes	T	No	574	512	512	No	256	No	14.764	0.402	4.77
cIRM+SA_T+F	Yes	T+F	No	578	512	512	No	256	No	15.011	0.723	8.59
cIRM+SA_T+BS	Yes	T	Yes	575	512	512	No	256	No	14.859	0.397	4.71
FLB: A (512/64/64)	Yes	T	No	574	512	64	64	64	64	14.048	0.403	1.19
FLB: A (256/32/128)	Yes	T	No	574	256	32	128	32	128	13.689	0.442	0.65
FLB: A (128/8/64)	Yes	T	No	574	128	8	64	8	64	11.322	0.692	0.25
FLB: A+B	Yes	T	No	574	256	32	128	32	128	13.538	0.468	0.69
FLB: A+B+C	Yes	T	No	574	256	32	128	32	128	13.204	0.577	0.85

Table 1. Separation performance of the accompaniment part for different experimental settings. The unit of segment and chunk size is the number of frames, where 8 frames are about 0.21 seconds in our setting. The three numbers of "FLB: A" are used to denoted different sizes of training segment, training chunk, and look back chunk. **Seg**: segment. **LBC**: look back chunk.

3.3. Results

Separation performance of the accompaniment part for different experimental settings is shown in Table 1, where the segment and chunk sizes are presented in the number of frames. The first four rows of Table 1 show the settings and evaluation results for the raw MMDenseNet, the MMDenseNet with Wiener filter, the MMDenseNet uses magnitude mask as output, and our improvement using cIRM. After invoking these improvements mentioned in Section 2.2, SDRs are significantly increased (from 11.162 to 13.872, 13.555, or 13.951), and the model using cIRM achieves highest SDR. On the other hand, since our modifications only change the output form (except the one with Wiener filter, which requires larger computational resources), the RTFs remain similar to the raw MMDenseNet.

By listening to the separated audio of these models, we found that outputs of the MMDenseNet uses magnitude mask as output sounds better than that of the raw MMDenseNet, and outputs of the MMDenseNet with Wiener filter sounds better than that of the MMDenseNet uses magnitude mask as output. Outputs of our improvement using cIRM and the MMDenseNet with Wiener filter sounds similar, but high frequency noise can be perceived in the output of our improvement using cIRM.

The fifth to seventh rows of Table 1 show that both using cIRM with self-attention and band-merge-split methods mentioned in Section 2.3 and 2.4 can further improve SDR (from 13.951 to 14.764, 15.011, or 14.859). Using self attention along both time and frequency axes improves SDR the most, but the RTF and latency are also higher, since the chunkwise self attention along the frequency axes requires larger computational resources than the self attention along the time axes. We also notice that the residual vocal sounds less after adding self attention. Two of these configurations, cIRM+SA_T and cIRM+SA_T+BS, achieve similar SDRs and RTFs. Despite the fact that the SDR of the cIRM+SA_T+BS is slightly better than that of cIRM+SA_T, we choose cIRM+SA_T for following experiments for the sake of implementation convenience. Besides, we test cIRM+SA_T using a shorter test chunk of 64 frames, and the SDR decreases from 14.764 to 12.776, showing that we cannot reduce the latency while still maintaining comparable separation quality by only shortening the test chunk.

Last five rows represent different settings and evaluation results for feature look back mentioned in Section 2.5. The three numbers of "FLB: A" are used to denoted different sizes of training segment, training chunk, and look back chunk. For the three settings of "FLB: A", the results show that when the sizes of training segments and chunks decrease, the SDRs also decrease, verifying out intuition in Section 2.5. However, these three SDRs (14.048, 13.689, and 11.322) are still higher than that of the raw MMDenseNet, showing that using feature look back is possible to reduce that latency while maintaining separation quality. Besides, with fixed sizes of training segments and chunks (256 and 32), and as more look back connections are used, the SDRs show a slight downward trend (13.689, 13.538, and 13.204), while the RTFs exhibit a rising trend. These results are caused by the fact that using more look back connections leads to increased computation in the network, and indicate that using shallow information together with deep information may not help increasing the separation quality. Finally, compared to the previous settings, all these FLB settings (except "FLB: A (128/8/64)") achieve

similar SDRs and listening perspective while significantly decreasing latencies, verifying the effectiveness of our proposed feature look back.

4. CONCLUSIONS AND FUTURE WORK

This paper uses the complex ideal ratio mask, self-attention, band-merge-split method, and feature look back to improve MMDenseNet for real-time application. Experimental results show that our method can significantly decrease the RTF and optimal latency while achieving similar SDRs compared to previous study.

Several directions for immediate future work are underway. Currently, the input spectrogram is cut into two subbands, but since the importance of subbands may be different [5], we could cut the input spectrogram into more subbands and investigate different ways of merging them. Moreover, although this study focuses on accompaniment separation due to application requirements, we could also investigate the performance of our model for different source types.

5. REFERENCES

[1] Divyesh G Rajpura, Jui Shah, Maitreya Patel, Harshit Malaviya, Kirtana Phatnani, and Hemant A Patil, "Effectiveness of transfer learning on singing voice conversion in the presence of background music," in *2020 International Conference on Signal Processing and Communications (SPCOM)*. IEEE, 2020, pp. 1–5.

[2] Xiaoxue Gao, Chitralekha Gupta, and Haizhou Li, "Music-robust automatic lyrics transcription of polyphonic music," in *Proceedings of the 19th Sound and Music Computing Conference*, 2022, pp. 325–332.

[3] Yuzhou Liu, Balaji Thoshkahna, Ali Milani, and Trausti Kristjansson, "Voice and accompaniment separation in music using self-attention convolutional neural network," *arXiv preprint arXiv:2003.08954*, 2020.

[4] Simon Rouard, Francisco Massa, and Alexandre Défossez, "Hybrid transformers for music source separation," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[5] Naoya Takahashi and Yuki Mitsufuji, "Multi-scale Multi-band Densenets for Audio Source Separation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2017, pp. 21–25.

[6] Qiuqiang Kong, Yin Cao, Haohe Liu, Keunwoo Choi, and Yuxuan Wang, "Decoupling magnitude and phase estimation with deep resunet for music source separation," *arXiv preprint arXiv:2109.05418*, 2021.

[7] Yi Luo and Jianwei Yu, "Music Source Separation With Band-Split RNN," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[8] Antoine Liutkus, Fabian-Robert Stöter, Zafar Rafii, Daichi Kitamura, Bertrand Rivet, Nobutaka Ito, Nobutaka Ono, and Julie Fontecave, "The 2016 signal separation evaluation campaign," in *Latent Variable Analysis and Signal Separation: 13th International Conference, LVA/ICA 2017, Grenoble, France, February 21-23, 2017, Proceedings 13*. Springer, 2017, pp. 323–332.

[9] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[10] Norbert Wiener, *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*, The MIT press, 1949.

[11] Zhong-Qiu Wang, Samuele Cornell, Shukjae Choi, Younglo Lee, Byeong-Yeol Kim, and Shinji Watanabe, "Tf-gridnet: Making time-frequency domain models great again for monaural speaker separation," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[12] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung, "Pointwise convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 984–993.

[13] Mikhail S Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V Sapunov, "Memory transformer," *arXiv preprint arXiv:2006.11527*, 2020.

[14] Wangyou Zhang, Kohei Saijo, Zhong-Qiu Wang, Shinji Watanabe, and Yanmin Qian, "Toward universal speech enhancement for diverse input conditions," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–6.

[15] Emmanuel Vincent, Rémi Gribonval, and Cédric Févotte, "Performance measurement in blind audio source separation," *IEEE transactions on audio, speech, and language processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[16] FR Stöter and A Liutkus, "museval 0.3. 0," 2019.

[17] Zafar Rafii, Antoine Liutkus, Fabian-Robert Stöter, Stylianos Ioannis Mimilakis, and Rachel Bittner, "The MUSDB18 corpus for music separation," Dec. 2017.