Audio Engineering Society

Convention Paper 10508

Presented at the 151st Convention
2021 October, Online

A Neural Beamforming Front-End for Distributed Microphone Arrays

Jonathan D. Ziegler[1, 2], Leon Schröder[1], Andreas Koch[1], and Andreas Schilling[2]

[1]*Stuttgart Media University, Institute for Applied Artificial Intelligence, Nobelstr. 10, 70569 Stuttgart, Germany*
[2]*Eberhard Karls Univerity, Institute for Visual Computing, Sand 14, 72076 Tübingen, Germany*

Correspondence should be addressed to Jonathan D. Ziegler (zieglerj@hdm-stuttgart.de)

ABSTRACT

Robust real-time audio signal enhancement increasingly relies on multichannel microphone arrays for signal acquisition. Sophisticated beamforming algorithms have been developed to maximize the benefit of multiple microphones. With the recent success of deep learning models created for audio signal processing, the task of Neural Beamforming remains an open research topic. This paper presents a Neural Beamformer architecture capable of performing spatial beamforming with microphones randomly distributed over very large areas, even in negative signal-to-noise ratio environments with multiple noise sources and reverberation. The proposed method combines adaptive, nonlinear filtering and the computation of spatial relations with state-of-the-art mask estimation networks. The resulting End-to-End network architecture is fully differentiable and provides excellent signal separation performance. Combining a small number of principal building blocks, the method is capable of low-latency, domain-specific signal enhancement even in challenging environments.

1 Introduction

High-quality, noise-free audio has become of ever greater importance with increases in human-computer interaction, telecommunication, web conferencing, and modern pro-audio applications. Ease of communication without personal contact has become a vital part of modern society. The enhancement of signals with respect to specific signal components such as speech can be performed with a wide variety of methods [1]. By time-aligning and filtering a set of microphone signals with respect to a defined signal source, surrounding noise and reverberation can be attenuated in the output signal. One of the largest challenges for such beamformers is the determination of the correct Time Difference of Arrival (TDOA), especially for spontaneous, large aperture microphone arrays of unknown configuration [2]. In recent years, machine learning has had a large impact on audio signal processing, redefining the state of the art in many topics. The task of source separation can be approached in numerous ways, with adaptive, nonlinear filtering on individual audio channels being the most prominent and easily available to date. Recently, multichannel, deep-learning based array processing has become increasingly relevant. The ability for Neural Beamforming networks to detect correlations of domain-specific signal components robustly and to generate the appropriate spatial filters is of great value. Current systems can generally be categorized into two approaches, in which the spatial information is either precomputed analytically and fed into the beamforming network [3, 4], or multiple neural networks

are used independently to achieve source separation [5, 6, 7]. Although investigations into End-to-End approaches have been remarkably successful [8, 9, 10], some basic limitations remain. Both referenced approaches directly use convolutions as beamforming filters. While Sainath et al. use dedicated convolutional layers to generate static spatial filters resulting in learnable "look-directions", Luo et al. use Temporal Convolutional Networks [11] or Dual-Path RNN [12] for the adaptive estimation of filters based on a preprocessed reference channel, raw auxiliary channels and the cosine similarity. While other approaches seem promising [13, 14], no performance data on arbitrary or large-aperture arrays are available for reference. The method introduced in the following sections circumvents the aforementioned limitations by using iterative downsampling and upsampling elements to adaptively produce long beamforming filters. The architecture presents an efficient End-to-End Neural Beamformer for large-aperture arrays and is capable of processing array signals for microphone distances of over 110 m in real time, outperforming the examined baseline approaches.

2 Neural Beamforming

The approach to Neural Beamforming described in the following sections tackles the task of generating appropriate spatial beamforming filters via the encoding of audio signals and spatial information into a shared latent space from which the beamforming filters are generated. Additionally, adaptive filtering of the audio channels prior to the computation of spatial relations is incorporated to enable the system to filter the input signals specifically for the task of spatial analysis. The use of learnable filter elements prior to the computation of pairwise cross correlations creates the ability to compute domain-specific spatial filters, which can greatly increase the beamformer's robustness to noise and reverberation. In section 2.1, an appropriate signal model is presented, sections 2.2 and 3 describe the method and implemented network architectures in detail.

2.1 Problem Definition

Spatial beamforming can be achieved using M audio channels m_ν recorded by $M \geq 2$ transducers. The discrete, time-domain channels consist of I signal and J noise components:

$$m_\nu[t] = \sum_{i=1}^{I} \tilde{s}_i^\nu[t] + \sum_{j=1}^{J} \tilde{n}_j^\nu[t]. \tag{1}$$

Every signal $\tilde{s}_i^\nu[t]$ and noise $\tilde{n}_j^\nu[t]$ consist of the corresponding signal and noise sources $s_i[t-\delta_i^\nu]$ and $n_j[t-\delta_j^\nu]$, propagated from their respective source position i or j to the transducer ν. The propagation transformations introduce time-delays δ and are expressed as convolutions with corresponding impulse responses $h_{s_i}^\nu$ and $h_{n_j}^\nu$ [15]:

$$\tilde{s}_i^\nu[t] = \left(s_i * h_{s_i}^\nu\right)[t], \quad \tilde{n}_j^\nu[t] = \left(n_j * h_{n_j}^\nu\right)[t], \tag{2}$$

resulting in

$$m_\nu[t] = \sum_{i=1}^{I} \left(s_i * h_{s_i}^\nu\right)[t] + \sum_{j=1}^{J} \left(n_j * h_{n_j}^\nu\right)[t]. \tag{3}$$

In this case, the goal is to find additional h_i^ν that maximize s_i in the summed combination ς_i of all m_ν:

$$\varsigma_i[t] = \sum_{\nu=1}^{M} \left(m_\nu * h_i^\nu\right)[t]. \tag{4}$$

Finding optimal h_i^ν is difficult and computationally expensive. For a simplified approach, h_i^ν can be approximated by a time shift δ_i^ν and a Finite Impulse Response (FIR) filter $\tilde{h}_i^\nu$ of relatively short length.

2.2 Differentiable Adaptive Generalized Cross Correlation

For microphone arrays of known spatial distribution, finding the correct time shifts δ_i^ν can be solved either geometrically, if the desired beam direction is known, or by means of a localization method, such as Steered-Response Power Phase Transform (SRP-PHAT) [16]. For arrays of unknown spatial distribution, pairwise time delay estimation can be performed by means of the cross correlation ϕ, expressed with the $\star$ operator [17]:

$$\phi_{m_1,m_2}[\tau] = (m_1 \star m_2)[\tau] \tag{5}$$

$$= \sum_{t=-\infty}^{\infty} \overline{m_1[t]} m_2[t+\tau], \tag{6}$$

with $\overline{[\quad]}$ describing the complex conjugation operation. Using the convolution theorem, (6) can be expressed as

$$\phi_{m_1,m_2}[\tau] = \mathscr{F}^{-1}\left\{\overline{\mathscr{F}\{m_1\}} \cdot \mathscr{F}\{m_2\}\right\}[\tau], \tag{7}$$

with $\mathscr{F}\{\quad\}$ representing the Fourier transform and $\mathscr{F}^{-1}\{\quad\}$ representing the inverse Fourier transform. Applying Phase Transform weighting leads to the computation of the Generalized Cross Correlation with Phase Transform weights (GCC-PHAT) [18]:

$$\phi^g_{m_1,m_2}[\tau] = \mathscr{F}^{-1}\left\{\frac{\overline{\mathscr{F}\{m_1\}} \cdot \mathscr{F}\{m_2\}}{\left|\overline{\mathscr{F}\{m_1\}} \cdot \mathscr{F}\{m_2\}\right|}\right\}[\tau]. \tag{8}$$

In the absence of interference and reverberation and for a single source s, the main peak of ϕ^g indicates the TDOA of the source with respect to the two input channels:

$$\delta^v = \underset{\tau}{\mathtt{arg\,max}}\left\{\phi^g_{m_0,m_v}[\tau]\right\}. \tag{9}$$

In real-world scenarios with multiple target and noise sources, TDOA estimation becomes unreliable, especially when using short audio buffers required for real-time applications. Addressing this problem from a data-driven perspective can improve the performance with domain knowledge.

Performing time alignment of the individual microphone signals within a neural network is a nontrivial task. As (9) is not differentiable, it cannot be incorporated into an End-to-End network architecture. Instead, spatial filters h^v_i are internally generated by the model, using latent representations of the input signals and the respective ϕ^g vectors. The generated filters are then applied to the microphone signals. While the process of Delay-and-Sum (DS) beamforming can be implemented in a strictly analytical way, adaptive pre-filtering and nonlinear pattern enhancement greatly improve performance over conventional methods. Additionally, the spatial relations for domain-specific signal classes can be computed, while (9) strictly extracts the correlation of the signal source with the highest sound pressure level found in the recorded signals.

3 Network Architecture

The main principle of the proposed architecture is to filter and synchronize multiple channels prior to multichannel mask estimation. GCC on adaptively filtered

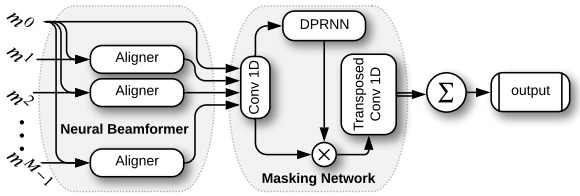

Fig. 1: Multiple audio channels are passed through the Neural Beamformer for synchronization. After spatial filtering, a linear encoder transforms the time-domain audio into a higher-dimensional feature vector. Then, a DPRNN mask approximation network generates channel-specific, per-sample feature-space amplitude masks, which are applied to the respective channels of encoded audio. After decoding the output to time-domain audio signals and summing the result, the model outputs a single-channel time-domain audio signal with the same buffer length as the reference input channel.

input signals is internally used by the network to extract feature-dependent, domain-specific spatial relations between channel pairs. Figure 1 shows the top-level model architecture, developed using the TensorFlow and Keras frameworks. A total of M microphones, with one defined reference channel m_0 are processed. Prior to multichannel mask estimation, every pair of signals m_0 and m_v is passed to an Aligner block which is discussed in detail in section 3.1 and can be seen in Figure 2. In the following sections, the individual components and the motivation behind the design choices are discussed.

3.1 Channel Synchronization - Aligner

One important requirement for the proposed method is the ability to extract the spatial relations of domain-specific signal components. Signal classes, for example *speech*, are to be enhanced in the signal prior to the computation of spatial relations. This enables the model to better make use of beamforming capabilities in environments that contain high levels of interference and noise. To achieve this, adaptive, nonlinear filtering of the input signals is implemented. The signals are encoded using EncodeWaveform blocks (subsection 3.3) and passed to FilterBlocks (subsection 3.2) for masking and filtering. The filtered signals are correlated to extract the spatial relation of the channels with respect to the desired signal components. The correlation

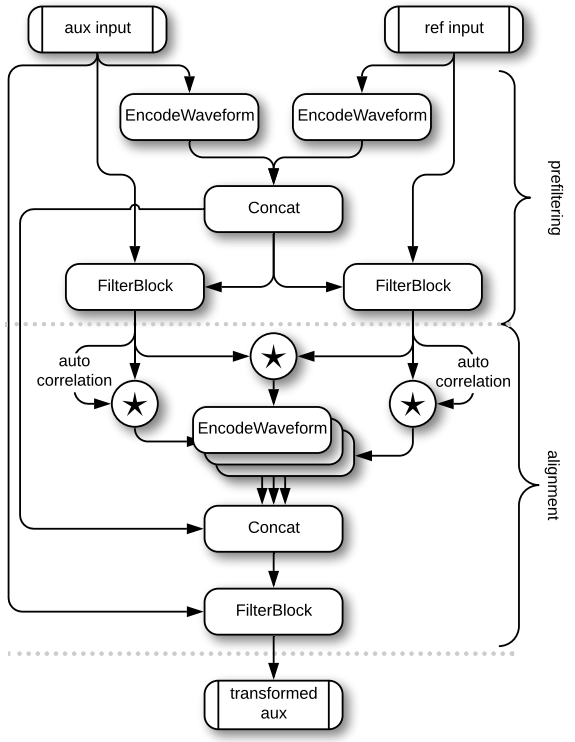

Fig. 2: Aligner architecture: Individual auxiliary and reference microphone pairs are processed with respect to the reference microphone. Adaptive FilterBlocks learn domain-specific signal components which are enhanced before spatial relations are computed for the filtered signals via cross- and autocorrelations. The encoded spatial vector is then used to generate multiplicative and convolutional filters in the final FilterBlock.

vectors are then encoded and concatenated with the latent vectors of both input channels. This expanded latent vector is then passed to an additional FilterBlock in combination with the original auxiliary input. In this block, channel synchronization and masking are performed to generate the transformed auxiliary signal.

3.2 Filtering and Masking - FilterBlock

The FilterBlock, shown in Figure 3, combines two main filtering approaches, namely convolutive filters and per-sample filter masks. Two individual GenerateWaveform blocks are used to generate filter vectors

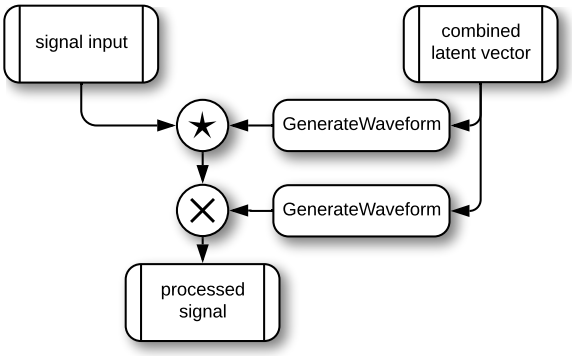

Fig. 3: The FilterBlock uses two GenerateWaveform blocks to generate FIR filters and amplitude masks and applies them to an input signal.

from the latent vector input. The filters are then applied via convolution and element-wise multiplication to the input signal. This flexible architecture provides capabilities for masking, FIR-filtering, scaling, and any desired combination of the aforementioned methods. For the configuration used to generate the results presented in this paper, the prefiltering blocks within the Aligner only use the masking components. The alignment filter generation exclusively relies on convolutive filtering to compensate spatial propagation and enhance the signal's spectral content.

3.3 Encoder - EncodeWaveform

To accommodate variable signal lengths with the same general architecture, the EncodeWaveform blocks, shown in Figure 4, are constructed using an iterative core, highlighted in gray. After normalization to zero mean and unit variance, activations (abbreviated with Act in the corresponding visualizations) and a multiplicative gate are created using convolutional layers feature space[19]. After passing through Layer Normalization, the output and the previous input are concatenated and passed to a convolutional downsampling layer, which serves as the input of the next iteration [20]. Once the required number of iterations has been performed, a final dense layer transforms the activations into the latent vector, which is then concatenated with the standard deviation and the mean of the input signal.

3.4 Decoder - GenerateWaveform

The GenerateWaveform blocks, shown in Figure 5, present the inverse operation of the EncodeWaveform.

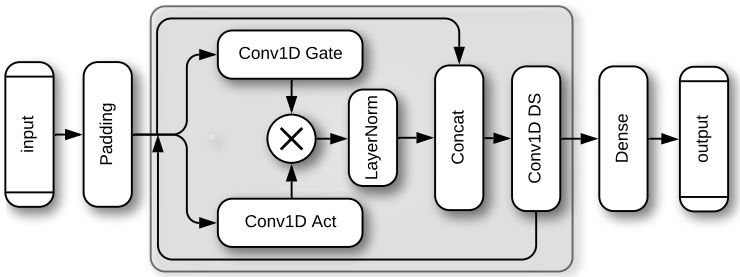

Fig. 4: The EncodeWaveform block uses gated and strided convolutions to perform encoding of signals to a defined latent size. Downsampling is performed by the strided convolutions and indicated by DS.

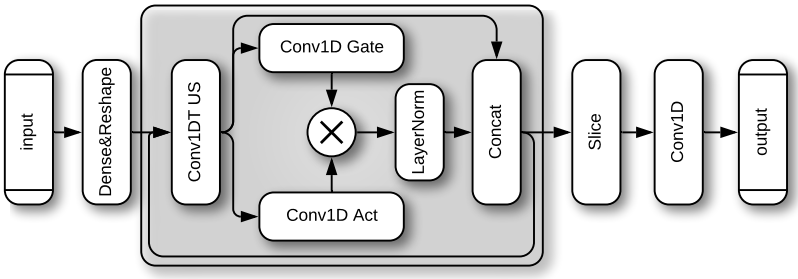

Fig. 5: The GenerateWaveform block uses gated and transposed convolutions to generate filters and amplitude masks from latent signal representations. Upsampling is performed by the transposed convolutions as indicated with US.

Such blocks can be used to create filter masks and spatial filters. Although the general architecture is identical for all applications, the iterative core enables a variable definition of the desired output size.

3.5 Mask Estimator - Dual-Path RNN

The Dual-Path RNN (DPRNN) architecture by Luo et al. generates state-of-the-art signal separation results for single-channel applications [12]. The approach splits the time-domain input buffers into smaller chunks, which are stacked to create input volumes. These volumes are then processed with separate bidirectional LSTM layers operating on the time and chunk axes. Additional linear convolution and transposed convolution layers encase the model and provide a learnable, time-domain base for the signal separation process. As shown in Figure 1, the synchronized array channels are individually processed by a DPRNN subnetwork and the resulting amplitude masks are applied to the signals prior to summation.

4 Experimental Setup

4.1 Training Data

Test and training data were created by simulating virtual acoustic environments [21, 22]. Room geometries were synthesized ranging from 8 m to 115 m per dimension, focusing on direct propagation and early reflections by restricting room simulation to third-order processing. One *signal*, a random number from three to seven *noise* sources, and a fixed number of microphones depending on the network configuration were randomly placed in the synthetic virtual environment. For training, 550000 multichannel buffers were randomly sampled from 550000 virtual recordings. Each buffer contains a reference channel, spanning 4096 samples and M-1 additional microphone signals of the same length, preceded by a context window of 12288 samples. The validation set contains 55000 buffers sampled correspondingly, using previously unseen source data. For the *signal* class, speech recordings from the VCTK corpus were used. *Noise* samples were extracted from the ESC50 corpus [23, 24]. All data are sampled at 48 kHz.

Conv Chanels	Conv Kernel Size	LSTM Features	Chunk Size	DPRNN Blocks	Buffer Size (Samples)
128	20	128	128	6	4096

Table 1: Parameters of the DPRNN Sub-Network.

Latent Size	US & DS Stride	US & DS Kernel Size	Gated Conv Kernel Size	Aligner Filter Length
32	8	8	5	16384

Table 2: Parameters of the Neural Beamformer Sub-Network.

4.2 Training Process

The model was trained using a MSE loss comparing the target signal recorded by the reference channel and the model output. Optimization was performed using the Ranger optimization algorithm [25, 26]. Standard Adam and Lookahead parameters were used, combined with a learning rate of 10^{-3}, a warm-up period of 500 steps and a learning rate decay of 4500 steps to reach the final rate of 10^{-4}. The training process was enhanced with channel dropout in which individual input channels were randomly set to 0 or filled with uncorrelated audio of the same signal statistics. The motivation of this dropout was to force the network to learn to completely reject individual input channels if necessary.

4.3 Model Configuration

Detailed information on the configuration chosen for the individual elements of the network can be referenced in Tables 1 and 2. Using this setup, the single-channel DPRNN contains approximately 1.4M parameters. The conversion to a multi-channel separation network using the Neural Beamformer front-end adds 347000 parameters, thus introducing an increase in model size of 25 % to 1.75M. Although the increase in model complexity is fairly modest, processing multiple channels with this shared architecture significantly increases the model runtime. When comparing the five-channel combination of Neural Beamformer and DPRNN with a single-channel DPRNN, the inference time of the model for a 85 ms buffer[1] increased from 32 ms to 80 ms on a single GPU.

5 Results

As a baseline, single- and multichannel versions of the same DPRNN architecture as used in the Neural Beamformer were trained (SC-DPRNN and MC-DPRNN). Additionally, Oracle and GCC-PHAT-based Delay-and-Sum beamforming, and combinations of GCC-PHAT and Oracle beamforming with the single channel DPRNN are referenced (Oracle, GCC-PHAT, GCC-DPRNN, Oracle-DPRNN). Overall separation performance is monitored by means of SDRi, the improvement of the Signal-to-Distortion Ratio, compared to the reference receiver, using [27]. In Table 3, an evaluation of the SDRi of the proposed method using five channels is presented under varying conditions, compared to the baseline approaches. The methods were compared on ten 20 s room simulations per configuration, resulting in a total of 40 min of audio used for evaluation. Audio material, room dimensions, and both microphone and source positions were randomly sampled for each iteration. During inference, each 20 s example is converted to blocks of 4096 + 12288 samples with 50 % overlap and passed to the individual processors. The resulting signals are (time-domain) Hann-windowed and recombined for evaluation. Even though the single-channel version of the DPRNN performs exceptionally well, applying the approach to multiple channels without spatial alignment results in negative SDRi (MC-DPRNN not shown in Table 3). As the Beamformer is trained in an End-to-End fashion, extracting this model component and evaluating the signal separation performance without the masking network would be misleading and thus has been omitted. Although the NBF-DPRNN model provides a relatively modest improvement in SDRi over the single-channel approach, the subjective reduction of artifacts and low-frequency residual noise is quite noticeable. The combination of Oracle beamforming with the DPRNN masking architecture shows that improved signal synchronization can provide a significant increase in overall model performance and future work will investigate improvements in the ability of the Beamformer components. Audio examples can be found at `https://zieglerj.home.hdm-stuttgart.de/nbf.html`.

[1] This time refers to the 4096 samples at 48 kHz sampling rate without the 12288 samples context provided to the network in the case of the Neural Beamformer.

	SNR	SDRi in dB			
	in dB	28 m	63 m	110 m	159 m
	-6	-3.11 ± 2.31	-3.03 ± 2.40	-2.79 ± 2.81	-2.28 ± 2.84
GCC	0	-0.95 ± 1.51	-1.38 ± 1.98	-1.94 ± 2.37	-2.04 ± 2.48
	6	-1.3 ± 1.10	-0.56 ± 1.62	-0.86 ± 2.12	-1.41 ± 1.98
	-6	6.56 ± 1.14	6.25 ± 0.91	6.1 ± 1.41	5.67 ± 1.61
Oracle	0	6.65 ± 1.13	6.34 ± 1.18	5.25 ± 1.87	4.15 ± 2.41
	6	6.57 ± 1.06	6.04 ± 1.77	4.75 ± 2.81	3.36 ± 3.56
GCC	-6	4.24 ± 3.26	4.19 ± 3.46	4.14 ± 3.92	4.69 ± 3.55
–	0	4.84 ± 1.85	4.15 ± 2.17	3.8 ± 2.26	3.96 ± 2.51
DPRNN	6	0.28 ± 1.82	1.06 ± 1.70	1.32 ± 1.94	1.66 ± 1.57
SC	-6	9.75 ± 1.00	9.72 ± 0.82	9.44 ± 0.80	9.48 ± 0.93
–	0	11.05 ± 1.88	11.08 ± 1.88	10.94 ± 1.86	10.86 ± 1.82
DPRNN	6	6.50 ± 1.32	6.46 ± 1.29	6.39 ± 1.26	6.13 ± 1.29
NBF	-6	10.19 ± 1.15	10.19 ± 1.13	9.96 ± 1.24	9.87 ± 1.18
–	0	11.58 ± 2.16	11.70 ± 2.00	11.59 ± 1.99	11.48 ± 2.12
DPRNN	6	7.12 ± 1.39	7.21 ± 1.27	7.14 ± 1.22	6.91 ± 1.35
Oracle	-6	14.66 ± 1.97	14.64 ± 1.95	14.27 ± 1.74	13.92 ± 2.07
–	0	14.78 ± 1.90	14.34 ± 1.57	13.42 ± 1.83	12.73 ± 2.07
DPRNN	6	8.77 ± 1.42	8.37 ± 1.56	7.75 ± 1.70	7.18 ± 1.72

Table 3: Performance Comparison of the Neural Beamformer in dB SDRi under variation of maximum microphone distance and mixture SNR at the reference channel.

6 Discussion

6.1 Processing of Large Time Delays

In order to operate in real time, buffers of 4096 samples were chosen. This presents a fundamental challenge for time-delays of over 85 ms, or distances of over 29 m. By inverting the application of the described model and focusing on noise modeling at the reference microphone instead of signal enhancement, the context vector of 12288 samples can be used more effectively. Beamforming is performed on the noise, which is more probable to have been recorded by other microphones before reaching the reference receiver and thus is captured in the context buffers. Signal enhancement is then performed by subtracting the modelled noise from the reference microphone signal. In this configuration, the possible delay compensation is only restricted by the microphone configuration and the chosen length of the context vector, which in this case contains a total of 16384 samples, resulting in a possible delay compensation of 341 ms, or 116 m. Even though the largest maximum microphone distance in Table 3 is 159 m, most examples are within the required distance as room dimensions, as well as microphone and source positions are uniformly sampled.

6.2 Training with Simplified Simulated Data

As mentioned in sections 4.1, data simulation was performed without diffuse reverberation. As previously stated, the exact inference of optimal impulse responses h_i^v is extremely complex, preventing reliable training convergence of the proposed model. Concentrating on the main contributing factors during training and excluding the task of explicit dereverberation presents an option for efficient and reliable training of Neural Beamformers. Inference experiments were performed showing that models trained with reduced simulation complexity are capable of performing well on data that contains a wide range of reverberation levels.

7 Conclusion

This paper presents a physics-informed and fully differentiable front-end for multichannel array processing, aimed at extracting domain-specific signals from a noisy mixture. Combined with mask estimation models, the high accuracy and short inference times enable the system to be used in real time applications. The proposed method outperforms the examined baseline approaches and provides a fully End-to-End neural beamforming network architecture, capable of processing microphone array signals with microphone distances of over 110 m.

Acknowledgments

This research was in part funded by the *Zentrales Innovationsprogramm Mittelstand*, a grant from the *Bundesministerium für Wirtschaft und Energie*.

References

[1] Benesty, J., Cohen, I., and Chen, J., *Fundamentals of Signal Enhancement and Array Signal Processing*, John Wiley & Sons Singapore Pte. Ltd, Singapore, 2017, ISBN 978-1-119-29313-2 978-1-119-29312-5, doi:10.1002/9781119293132.

[2] Ying Yu and Silverman, H. F., "An improved TDOA-based location estimation algorithm for large aperture microphone arrays," in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, 2004.

[3] Gu, R., Chen, L., Zhang, S.-X., Zheng, J., Xu, Y., Yu, M., Su, D., Zou, Y., and Yu, D., "Neural spatial filter: target speaker speech separation assisted with directional information," in *Interspeech 2019*, pp. 4290–4294, ISCA, 2019, doi: 10.21437/Interspeech.2019-2266.

[4] Qian, K., Zhang, Y., Chang, S., Yang, X., Florencio, D., and Hasegawa-Johnson, M., "Deep learning based speech beamforming," in *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5389–5393, 2018, doi:10.1109/ICASSP.2018.8462430.

[5] Wang, Z.-Q. and Wang, D., "All-neural multichannel speech enhancement," in *Interspeech 2018*, pp. 3234–3238, ISCA, 2018, doi:10.21437/Interspeech.2018-1664.

[6] Koyama, Y. and Raj, B., "W-Net BF: DNN-based Beamformer Using Joint Training Approach," in *arXiv:1910.14262 [cs, eess]*, 2019.

[7] Yoshioka, T., Chen, Z., Liu, C., Xiao, X., Erdogan, H., and Dimitriadis, D., "Low-latency speaker-independent continuous speech separation," in *ICASSP 2019-2019 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 6980–6984, 2019.

[8] Sainath, T. N., Weiss, R. J., Wilson, K. W., Narayanan, A., Bacchiani, M., and Senior, Andrew, "Speaker location and microphone spacing invariant acoustic modeling from raw multichannel waveforms," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pp. 30–36, IEEE, Scottsdale, AZ, USA, 2015, ISBN 978-1-4799-7291-3, doi: 10.1109/ASRU.2015.7404770.

[9] Sainath, T. N., Weiss, R. J., Wilson, K. W., Li, B., Narayanan, A., Variani, E., Bacchiani, M., Shafran, I., Senior, A., Chin, K., Misra, A., and Kim, C., "Multichannel signal processing with deep neural networks for automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 25(5), pp. 965–979, 2017, ISSN 2329-9290, 2329-9304, doi: 10.1109/TASLP.2017.2672401.

[10] Luo, Y., Han, C., Mesgarani, N., Ceolini, E., and Liu, S.-C., "FaSNet: Low-latency adaptive beamforming for multi-microphone audio processing," in *2019 IEEE automatic speech recognition and understanding workshop (ASRU)*, pp. 260–267, 2019.

[11] Luo, Y. and Mesgarani, N., "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, 27(8), pp. 1256–1266, 2019.

[12] Luo, Y., Chen, Z., and Yoshioka, T., "Dual-path RNN: efficient long sequence modeling for time-domain single-channel speech separation," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 46–50, 2020.

[13] Gu, R., Wu, J., Zhang, S.-X., Chen, L., Xu, Y., Yu, M., Su, D., Zou, Y., and Yu, D., "End-to-End Multi-Channel Speech Separation," in *arXiv:1905.06286 [cs, eess]*, 2019.

[14] Wu, J., Chen, Z., Li, J., Yoshioka, T., Tan, Z., Lin, E., Luo, Y., and Xie, L., "An End-to-End Architecture of Online Multi-Channel Speech Separation," in *Interspeech 2020*, pp. 81–85, ISCA, 2020, doi:10.21437/Interspeech.2020-1981.

[15] Benesty, J., Jingdong, C., and Huang, Y., *Microphone Array Signal Processing*, Springer Berlin Heidelberg, 2008, ISBN 978-3-540-78612-2.

[16] DiBiase, J. H., *A High-Accuracy, Low-Latency Technique for Talker Localization in Reverberant Environments Using Microphone Arrays*, Ph.D. thesis, Brown University, 2000.

[17] Azaria, M. and Hertz, D., "Time delay estimation by generalized cross correlation methods," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2), pp. 280–285, 1984, doi:10.1109/TASSP.1984.1164314.

[18] Knapp, C. and Carter, G., "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4), pp. 320–327, 1976, doi:10.1109/TASSP.1976.1162830.

[19] Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K., "WaveNet: a generative model for raw audio," in *arXiv:1609.03499 [cs]*, 2016.

[20] Ba, J. L., Kiros, J. R., and Hinton, G. E., "Layer normalization," *arXiv:1607.06450 [cs, stat]*, 2016.

[21] Allen, J. B. and Berkley, D. A., "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, 65(4), pp. 943–950, 1979.

[22] Scheibler, R., Bezzam, E., and Dokmanić, I., "Pyroomacoustics: A Python package for audio room simulations and array processing algorithms," 2018.

[23] Veaux, C., Yamagishi, J., and MacDonald, K., "CSTR VCTK corpus: English multi-speaker corpus for CSTR voice cloning toolkit," *University of Edinburgh. The Centre for Speech Technology Research (CSTR)*, 2017.

[24] Piczak, K. J., "ESC: dataset for environmental sound classification," in *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, pp. 1015–1018, ACM, New York, NY, USA, 2015, ISBN 978-1-4503-3459-4, doi:10.1145/2733373.2806390, event-place: Brisbane, Australia.

[25] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J., "On the variance of the adaptive learning rate and beyond," in *8th international conference on learning representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020.

[26] Zhang, M., Lucas, J., Ba, J., and Hinton, G. E., "Lookahead optimizer: k steps forward, 1 step back," in H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pp. 9597–9608, Curran Associates, Inc., 2019.

[27] Raffel, C., McFee, B., Humphrey, E. J., Salamon, J., Nieto, O., Liang, D., and Ellis, D. P. W., "MIR_EVAL: a transparent implementation of common MIR metrics." in H.-M. Wang, Y.-H. Yang, and J. H. Lee, editors, *ISMIR*, pp. 367–372, 2014.