Towards an efficient computation of masks for multichannel speech enhancement

Louis Delebecque, Romain Serizel, Nicolas Furnon

Towards an efficient computation of masks for multichannel speech enhancement

Louis Delebecque, Romain Serizel, Nicolas Furnon
Université de Lorraine, CNRS, Inria, Loria
F-54000 Nancy, France
{firstname.lastname}@loria.fr

***Abstract*—Most of recent advances in speech enhancement (SE) have been enabled by the use of complex deep neural network (DNN) architectures. Although these results are convincing, they are not yet applicable in small wearable devices like hearing aids. In this paper, we propose a DNN-based SE which benefits from the spatial information to simplify the requirements of the DNN architecture. We show that the DNN inference is the most time and energy consuming step and we simplify the architecture of a convolutional recurrent neural network by removing its recurrent layer. This achieves comparable performance to the initial architecture, while reducing the processing time and energy consumption by a factor of 4.4.**

***Index Terms*—Fast inference, speech enhancement**

I. Introduction

Speech enhancement (SE) benefits a lot from spatial information captured by several devices : using several microphones embedded in one or several devices has shown to deliver improved performance over single-channel SE [1], [2]. In recent years, DNN-based solutions enabled a great progress in both single-channel [3], [4] and multichannel [5], [6] SE, but at the cost of increasingly complex DNN architectures, requiring powerful devices even at inference, which makes these solutions impractical in real life if they are to operate on small wearable devices like hearing aids.

Alleviating the computational cost of DNN-based SE solutions has been the focus of a number of research works and addressed with different approaches. One way of reducing the computational cost is to reduce the dimensionality of the input data, for example by considering low-dimensional features [7], [8] or by applying an element-selection method [9]. DNN compression [10] has also allowed for significant model reduction, without impacting too much, if at all, the model performance. Examples of model compression are network pruning [11], [12], weights and activations quantization [13], knowledge distillation into smaller network architectures [14], or a combination thereof. Fedorov et al. for example direclty learn a pruned structure by incorporating a penalty in the loss function, thus avoiding costly hyperparameter search [15]. Rather than compressing a complex DNN, the computational cost of a DNN can be alleviated by the design of a simple architecture. For example Sivaraman et al. accelerate the inference by selecting a smaller sub-network specialised in specific sub-conditions [16]. In a similar inference-acceleration spirit, Chen et al. stop the inference with an early-exit mechanism when the distance between the output of two successive layers remains below a given threshold [17].

Despite numerous solutions to reduce the computational cost of DNN in a SE context, these solutions suffer from several drawbacks. The first of them is that they barely exploit the spatial information, although this extra information can help reducing the complexity of DNNs. Besides, network pruning does not necessarily translate into faster learning and inference because most of the software and hardware implementations are not adapted to sparse structures [18]. Lastly, model compression, especially knowledge distillation, requires to fine-tune or even retrain the models, which can have a high cost, in particular when considering the search of hyperparameters.

In previous works, we introduced a distributed SE system for ad-hoc microphone arrays, called Tango [19], [20]. This solution exploits spatial information and relies partly on classical processing which allows respectively to provide rich information to DNNs and to alleviate its task.

In this paper, based on this system, we propose an efficient and fast SE solution, that is a first step towards a real-time implementation on embedded devices. The original convolutional recurrent neural network (CRNN) architecture of our DNNs is further simplified by a direct work on the network architecture, without compression, thus avoiding the process of retraining models. We perform a detailed ablation study to better dissociate the effects of all elements of our processing pipeline and report both time and energy usage of all these elements. We propose a closer study of the relevance of time information in our SE. Lastly, the importance of the recurrent layer in the CRNN is analysed and we show that this layer can be removed.

II. Problem formulation

A. Notations

In the following, signals will be considered in the short-time Fourier transform (STFT) domain, where time and frequency indices are dropped for the sake of conciseness. Bold lowercase letters will represent vectors. Bold uppercase

This work was made with the support of the French National Research Agency, in the framework of the project DiSCogs "Distant speech communication with heterogeneous unconstrained microphone arrays" (ANR-17-CE23-0026-01). Experiments presented in this paper were partially carried out using the Grid5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000).

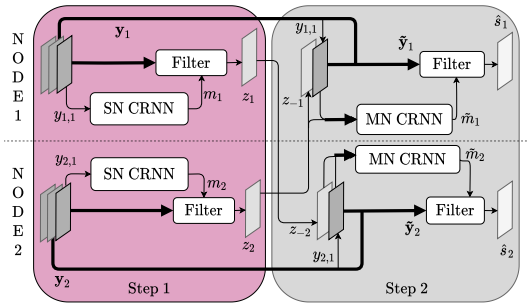

Fig. 1: Graphical representation of our distributed SE solution in a two-node case. Bold arrows represent multichannel signals, simple arrows represent single-channel signals.

letters will represent matrices. Regular letters will represent scalars. We consider K nodes of I_k microphones each. The i-th microphone of the k-th node records a noisy mixture $y_{k,i} = s_{k,i}+n_{k,i}$ according to an additive noise model, where $s_{k,i}$ and $n_{k,i}$ are respectively the target speech and the noise components recorded by the microphone. The signals recorded by node k are stacked in a vector $\mathbf{y}_k = [y_{k,1}, \cdots, y_{k,I_k}]^T$.

B. Distributed SE in ad-hoc microphone arrays

Tango SE system processes in two steps, represented in Figure 1. In the first step, a multichannel Wiener filter (MWF) is applied at each node on the local signals $\mathbf{y}_k$. To do this, a single-node CRNN (SN CRNN) is used to predict a time-frequency (TF) mask m_k out of the reference signal $y_{k,1}$. The TF mask is used to compute the spatial covariance matrices of the speech and noise required by the MWF :

$$\mathbf{R}_{\mathrm{s},k}(f) = \frac{1}{\mathcal{T}} \sum_{t=1}^{\mathcal{T}} \hat{\mathbf{s}}_k(f,t)\hat{\mathbf{s}}_k(f,t)^H$$

where $\mathbf{R}_{\mathrm{s},k}$ is the spatial covariance matrix of the speech; $\mathcal{T}$ is the number of STFT frames of the signals; $\cdot^H$ denotes the Hermitian transpose; $\hat{\mathbf{s}}_k$ is estimated as $\hat{\mathbf{s}}_k = m_k \cdot \mathbf{y}_k$. The noise covariance matrix is similarly computing with $\hat{\mathbf{n}}_k = (1 - m_k) \cdot \mathbf{y}_k$.

The local MWF $\mathbf{w}_{kk}$ is computed following the rank-1 generalized eigenvalue decomposition (GEVD) of the matrix pencil $\{\mathbf{R}_{\mathrm{y},k}, \mathbf{R}_{\mathrm{n},k}\}$ [21]. Filtering the mixture with this beamformer yields a so-called compressed signal $z_k = \mathbf{w}_{kk}^H \mathbf{y}_k$. The compressed signals are exchanged among nodes, so the node k receives $K-1$ compressed signals $\mathbf{z}_{-k}$: $\mathbf{z}_{-k} = [z_1, ..., z_{k-1}, z_{k+1}, ..., z_K]^T$. In the second step, a global MWF is applied on $\tilde{\mathbf{y}}_k = \left[\mathbf{y}_k^T,\ \mathbf{z}_{-k}^T\right]^T$, where a multi-node CRNN (MN CRNN) predicts a second TF mask $\tilde{m}_k$, this time out of the local reference signal $y_{k,1}$ and the received compressed signals $\mathbf{z}_{-k}$.

We showed that this algorithm could efficiently process the spatial information conveyed by the compressed signals and outperform an oracle voice activity detector (VAD)-based MWF [19]. We also showed in another study [20] that it performs comparatively well to FaSNet [22], while allowing for a trade-off between noise reduction and speech distortion, and relying on a much simpler DNN architecture.

III. Experimental setup

A. Datasets

The dataset used to train the DNNs and evaluate our proposed solution is the same as the one of our previous work [20]. It consists of simulations of shoebox-like rooms with one target source and one noise source randomly laid in the room. Four nodes of four microphones each are also randomly placed in the room. Dimensions of the room are chosen randomly within the following ranges : from 3 to 8 m for the length, from 3 to 5 meters for the width and from 2.5 to 3 m for the height.

All sources and nodes are distant of at least 50 cm of the closest source, node and wall. The speech material is taken from LibriSpeech [23]. The noise material is downloaded from Freesound [24]. It is split into two non-overlapping subsets of Freesound users for the training and testing sets[1]. Some speech-shaped noise was also used to train the DNN because it was shown to improve the robustness of the DNN [20]. The rooms were simulated with the Python toolbox Pyroomacoustics [25]. The source to interferences ratio (SIR) of the non-reverberated source signals is randomly taken between 0 dB and 6 dB. The reverberation time ranges from 300 ms to 600 ms. We created around 25 hours of training material (gathering both training and validation sets) and 2.7 hours of testing material.

B. Models parameters

All the signals are sampled at 16 kHz. The STFT is computed with a Hann window of 32 ms with an overlap of 16 ms. The CRNN architecture is composed of three convolutional layers followed by a recurrent layer and a fully-connected layer. The convolutional layers have 32, 64 and 64 filters, with kernel size 3×3 and stride 1×1. Each convolutional layer is followed by a batch normalisation and a maximum-pooling layer of kernel size 4×1 so that no pooling is applied over the time axis. The recurrent layer is a 256-unit GRU. The fully-connected layer has 257 units with a sigmoid activation function. The input of the model are the magnitudes of the STFT windows of 21 consecutive frames and the ground truth labels are the corresponding frames of the ideal ratio mask. At test time, only the middle frame of the predicted window is considered to estimate the mask, so sliding windows of the input are fed to the DNN. The mask of the whole signal is predicted before being used to enhance the speech in a batch mode.

C. Performance evaluation

Three metrics are used to evaluate the results: the SIR improvement, denoted as ΔSIR; the source to artifacts ratio (SAR); and the source to distortion ratio (SDR) [26]. The references needed to compute these metrics are the reverberated

[1]The noise dataset is available at https://zenodo.org/record/4019030.

Processing stage	Processing time	Energy consumed (Wh)
CRNN step 1	1722.3 ± 5.2	8.28 ± 0.08
CRNN step 2	1778.6 ± 6.7	8.52 ± 0.10
MWF step 1	24.5 ± 0.5	0.09 ± 0.03
MWF step 2	41.0 ± 0.7	0.16 ± 0.03

TABLE I: Computing time and energy consumed for each of the four stages in Tango, mean values and 95% confidence intervals over 3 measures.

noise and speech signals. The results reported in this work are the average over the whole evaluation dataset of the metrics computed on the node with the highest SIR among the four nodes of each evaluation configuration.

Power consumption and processing time are computed on the full evaluation dataset. To do this, Tango is run on 48 cores from Intel Xeon E5-2650 v4 CPUs. We choose to run the experiments on CPU under the assumption that GPU are not always available at runtime, in particular on embedded devices. The power consumption reported in this paper is computed using CodeCarbon [27].

IV. Results and discussion

In the following experiments, we propose to analyse to which extent we can reduce the CRNN complexity while maintaining the performance of the SE system. In a first set of experiments, we investigate the baseline system. In a second set of experiments, we propose several ways of directly reducing the DNN complexity and analyse their impact in terms of both SE performance and power consumption.

A. Baseline CRNN analysis

Tango is a rather complex algorithm involving two filtering steps. Each step in turn includes a masking stage based on a CRNN model and a filtering stage based on a rank-1 GEVD-MWF [21]. Note that the signal conversion from time domain to time-frequency domain is not taken into account here as reducing the complexity of this step is out of the scope of the paper.

Table I presents the computing time and the energy consumed for each of the four filter stages described above. The CRNN stages for both steps 1 and 2 are much more demanding than the MWF filtering stages. This is true for the processing time as well as the energy consumed during the computation. This large difference can partially be attributed to the fact the MWF is operated at a batch level (only one filter is computed for the whole segment), but is consistent with the results observed by other researchers [13]. In this particular operating mode, reducing the complexity of the models used to estimate the mask appears to be the most obvious way to reduce the overall processing time and power consumption. Note that as the processing time and energy consumption of the CRNN are similar for both steps, in the remainder of the paper, unless stated otherwise, we will present only the processing time and energy consumption for the CRNN at the second step (even though networks at both steps are always modified simultaneously).

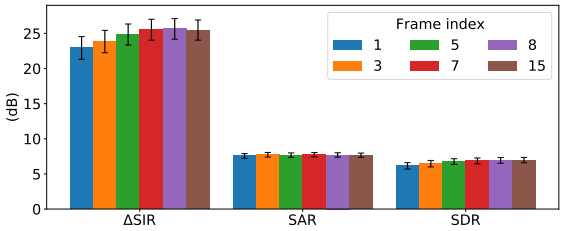

Fig. 2: SE performance when different output frames are considered by the GRU layer.

The baseline model includes a gated recurrent unit (GRU) recurrent layer between the convolutional layers and the last fully-connected layer of the network. Previous studies have reported that recurrent layers have a rather high computational cost for little impact on the final performance of the systems [28]. Here we propose to analyse the performance of Tango depending on the context that is effectively taken into account by the recurrent part of the network. Figure 2 presents the SE performance obtained with Tango when different output frames are considered by the GRU layer. Since the recurrent layer is unidirectional, taking its first output frame results in not using any context at test time. Taking the last frame corresponds to using the context of the past 14 frames[2].

The Figure 2 shows that the SIR improvement increases softly with the output frame index until reaching a constant value for the frame 7. However this improvement is not significant and other metrics present similar SE performance regardless of the output frame. Therefore, at runtime, the temporal context at the input of the GRU does not seem to be critical.

Since the GRU context does not provide any significant performance improvement at runtime, two simple options follow to reduce the power consumption and the latency of the CRNN. The first option is to use the model in a sequence-to-sequence mode, where all the 15 frames are outputted at once, which reduces the number of inferences needed to construct the TF mask. The second option is to simply use the first output frame, which reduces the latency. Not using the context at runtime does not mean that it is not necessary during training when the recurrent layer might learn something from the context. To verify this latter hypothesis we propose to retrain the CRNN but where only 1 frame is fed to the recurrent neural network (RNN) [3].

Figure 3 presents the performance obtained with Tango when considering different options to construct the TF mask. The method denoted Base-1 considers the first output frame of the baseline CRNN. Using this same notation, the baseline would be referred as Base-8. The method denoted Base-seq uses the baseline in a sequence-to-sequence mode, taking all the 15 output frames to construct the TF mask. The method

[2]The DNN input consists in 21 frames. Because of the 3×3 convolutional kernels, 15 frames remain at the input of the GRU layer.

[3]A third option would be to output the last 8 frames at once in a sequence-to-sequence mode in order to maximise SE performance with a reasonably low complexity.

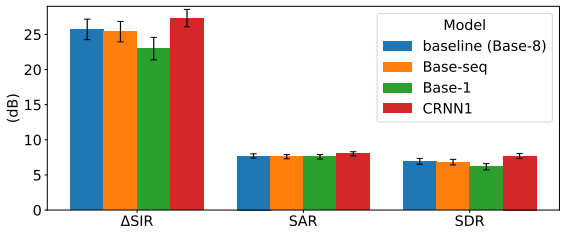

Fig. 3: SE performance of the baseline along with three simplified CRNN configurations.

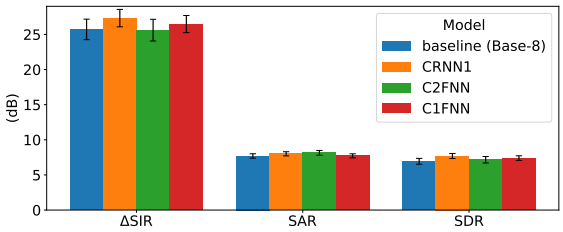

Fig. 4: SE performance when the GRU layer of the CRNN is replaced by a fully-connected layer (C2FNN) or by the identity (C1FNN).

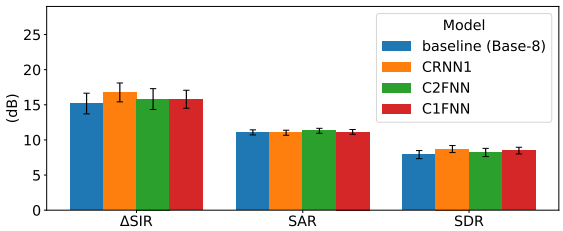

Fig. 5: SE performance when the masks predicted by the different DNNs at the second step are directly used as filters.

denoted CRNN1 retrains the baseline CRNN without any temporal context at the input of the recurrent part, so only 1 frame is outputted by the CRNN.

The performance obtained with Base-seq confirms the observation from the previous experiments (Figure 2) while the performance obtained with Base-1, compared to the one obtained with CRNN1, indicates that the temporal context at the input of the GRU is not exploited if this context differs from the one used at test time.

Table II presents the computing time and the energy consumed by each of the proposed alternatives. Compared to the original baseline consumption and processing time, values obtained for Base-seq and CRNN1 are reduced by a large margin. Although Base-seq provides the larger reduction, because of its sequence-to-sequence configuration, it also potentially introduces a larger processing latency. Therefore, in the remainder of the paper, we focus on models derived from CRNN1.

Network	Processing time	Energy consumed (Wh)
baseline	1778.6 ± 6.7	8.52 ± 0.10
Base-seq	195.0 ± 2.3	0.94 ± 0.02
CRNN1	404.2 ± 2.6	1.95 ± 0.05
C2FNN	397.4 ± 4.1	1.95 ± 0.03
C1FNN	401.1 ± 3.0	1.94 ± 0.01

TABLE II: Processing time and energy consumption of the CRNN stage at the second filtering step, mean values and 95% confidence intervals over 3 measures.

B. Simplified networks performance

Based on the previous experiments, given the limited impact of the temporal context of the CRNN, we propose to remove the recurrent layer. One solution is to simply remove the layer, and will be referred to as C1FNN. Another solution is to replace the recurrent layer by a fully connected layer with 256 units. It will be referred to as C2FNN. Figure 4 presents the SE performance for the baseline, CRNN1, C1FNN and C2FNN. Performance is similar for all approaches, which means that the recurrent layer could simply be removed to reduce the complexity. However, the impact of this simplification is not obvious in terms of processing time and energy consumption (see C1FNN compared to CRNN1 in Table II). This is due to the fact that, most of the resources of the network were allocated to the RNN that considers some temporal context (baseline). When no temporal context is considered (CRNN1), resources allocated to this same part are negligible. Compared to the baseline, reduction in processing time and consumption for C1FNN model is 4.4.

C. Robustness to mask estimation

Previous experiments showed that Tango is robust to changes in the models used for the mask estimation. In this section, we investigate the reason for this robustness. A first hypothesis is that the robustness comes from the fact that the masks are used to compute a MWF rather than directly used for masking. To check this hypothesis, we report in Figure 5 the performance when the TF masks predicted at the second step are directly used as filters, applied on the reference signal of each node. As reported in Figure 5, all networks perform similarly well when their output is used for masking. Therefore, our first hypothesis cannot explain the robustness of Tango to simpler models.

A second hypothesis is that the two-step mechanism of Tango brings robustness. The mask estimation at the second step heavily relies on the signals estimated at the first step [20]. To check this hypothesis, we report in Figure 6 the performance obtained at the first filtering step of Tango when masks are used to estimate a MWF (Fig. 6a) and when they are directly used a filters (Fig. 6b). The performance of the first step MWF is stable regardless of the network used during the mask estimation (Fig. 6a). However, when directly applying the estimated masks as filters (Fig. 6b), changes in the network architecture have a moderate impact on the performance. Compared to the baseline, masks obtained without using recurrent networks (C2FNN and C1FNN) introduce more artefact (lower SAR) while obtaining similar SIR improvement. The fact that the masks are not directly used but serve to compute a MWF and the two-step mechanism

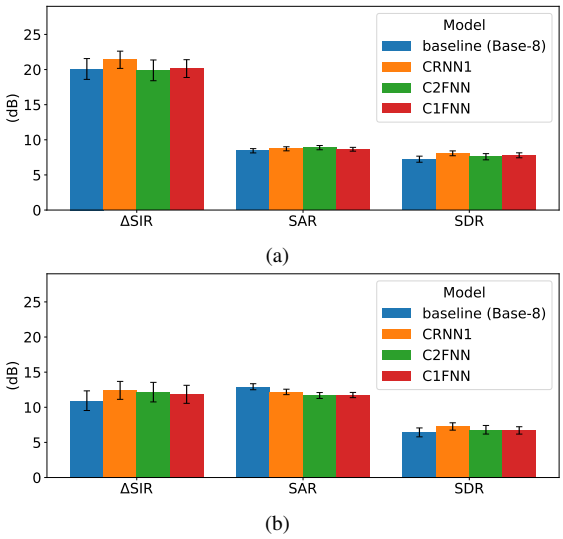

Fig. 6: SE performance at the first filtering step, when masks are used (a) to estimate a MWF or (b) directly as a filter.

of Tango bring therefore additional robustness. Even with a coarse TF mask prediction at the first step, the following MWF allows the generated signals to be sufficiently clean to allow simple models to predict accurate TF masks at the second step.

V. CONCLUSION

In this paper, we investigated several factors to simplify and accelerate the process of a distributed SE pipeline. To do this, we focused on the inference of CRNNs as they showed to contribute the most in the time and energy consumption. We showed that the temporal context at the input of the GRU layer of the CRNN is relevant neither for the training nor for the test. Based on this observation, we simplified the CRNN, removing its recurrent layer. Compared to the initial baseline, our simplified model reduces the processing time and energy consumption by a factor of 4.4. The pipeline of our SE system, composed of two filtering steps in which masks predicted by the DNN are used to compute a MWF, brings additional robustness to mask estimation. Future work could consist in analysing whether the batch-mode processing of the signals helps smoothing out the prediction errors of the CRNNs, allowing for coarse mask estimation without impacting the final SE performance.

REFERENCES

[1] M. Souden, J. Chen, J. Benesty, and S. Affes, "An integrated solution for online multichannel noise tracking and reduction," *IEEE/ACM TASLP*, vol. 19, no. 7, pp. 2159–2169, 2011.

[2] E. Ceolini, I. Kiselev, and S.-C. Liu, "Evaluating multi-channel multi-device speech separation algorithms in the wild: a hardware-software solution," *IEEE/ACM TASLP*, vol. 28, pp. 1428–1439, 2020.

[3] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM TASLP*, vol. 27, no. 8, pp. 1256–1266, 2019.

[4] 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," in *Proc. Interspeech 2020*, 2020, pp. 2472–2476.

[5] T. Ochiai, M. Delcroix, R. Ikeshita, K. Kinoshita, T. Nakatani, and S. Araki, "Beam-tasnet: Time-domain audio separation network meets frequency-domain beamformer," in *2020 ICASSP*. IEEE, 2020, pp. 6384–6388.

[6] Y. Koyama and B. Raj, "Exploring optimal DNN architecture for end-to-end beamformers based on time-frequency references," *arXiv preprint arXiv:2005.12683*, 2020.

[7] J.-M. Valin, "A hybrid DSP/deep learning approach to real-time full-band speech enhancement," in *2018 MMSP*. IEEE, 2018, pp. 1–5.

[8] G. S Bhat, N. Shankar, C. KA Reddy, and I. MS Panahi, "A real-time convolutional neural network based speech enhancement for hearing impaired listeners using smartphone," *IEEE Access*, vol. 7, pp. 78421–78433, 2019.

[9] C. Haruta and N. Ono, "A low-computational dnn-based speech enhancement for hearing aids based on element selection," in *2021 EUSIPCO*. IEEE, 2021.

[10] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.

[11] K. Tan and DL. Wang, "Compressing deep neural networks for efficient speech enhancement," in *2021 ICASSP*. IEEE, 2021, pp. 8358–8362.

[12] K. Tan, X. Zhang, and DL. Wang, "Deep learning based real-time speech enhancement for dual-microphone mobile phones," *IEEE/ACM TASLP*, vol. 29, pp. 1853–1863, 2021.

[13] L. Pfeifenberger, M. Zöhrer, G. Schindler, W. Roth, H. Fröning, and F. Pernkopf, "Resource-efficient speech mask estimation for multi-channel speech enhancement," *arXiv preprint arXiv:2007.11477*, 2020.

[14] X. Hao, X. Su, Z. Wang, Q. Zhang, H. Xu, and G. Gao, "SNR-Based Teachers-Student Technique For Speech Enhancement," in *2020 ICME)*. IEEE, 2020, pp. 1–6.

[15] I. Fedorov, M. Stamenovic, C. Jensen, L.-C. Yang, A. Mandell, Y. Gan, M. Mattina, and P. N. Whatmough, "TinyLSTMs: Efficient neural speech enhancement for hearing aids," *arXiv preprint arXiv:2005.11138*, 2020.

[16] A. Sivaraman and M. Kim, "Sparse mixture of local experts for efficient speech enhancement," *arXiv preprint arXiv:2005.08128*, 2020.

[17] S. Chen, Y. Wu, Z. Chen, T. Yoshioka, S. Liu, J. Li, and X. Yu, "Don't shoot butterfly with rifles: Multi-channel continuous speech separation with early exit transformer," in *2021 ICASSP*. IEEE, 2021, pp. 6139–6143.

[18] X. Zhou, Z. Du, S. Zhang, L. Zhang, H. Lan, S. Liu, L. Li, Q. Guo, T. Chen, and Y. Chen, "Addressing sparsity in deep neural networks," *IEEE TCADICS*, vol. 38, no. 10, pp. 1858–1871, 2018.

[19] N. Furnon, R. Serizel, I. Illina, and S. Essid, "Dnn-based distributed multichannel mask estimation for speech enhancement in microphone arrays," in *2020 ICASSP*. IEEE, 2020, pp. 4672–4676.

[20] N. Furnon, R. Serizel, S. Essid, and I. Illina, "Dnn-based mask estimation for distributed speech enhancement in spatially unconstrained microphone arrays," *IEEE/ACM TASLP*, vol. 29, pp. 2310–2323, 2021.

[21] R. Serizel, M. Moonen, B. Van Dijk, and J. Wouters, "Low-rank Approximation Based Multichannel Wiener Filter Algorithms for Noise Reduction with Application in Cochlear Implants," *IEEE/ACM TASLP*, vol. 22, no. 4, pp. 785–799, 2014.

[22] Y. Luo, C. Han, N. Mesgarani, E. Ceolini, and S.-C. Liu, "Fasnet: Low-latency adaptive beamforming for multi-microphone audio processing," in *2019 ASRU*. IEEE, 2019, pp. 260–267.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," *IEEE ICASSP*, pp. 5206–5210, 2015.

[24] F. Font, G. Roma, and X. Serra, "Freesound technical demo," *ACM International Conference on Multimedia (MM'13)*, pp. 411–412, 2013.

[25] R. Scheibler, E. Bezzam, and I. Dokmanic, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," *IEEE ICASSP*, Apr 2018.

[26] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE TASLP*, vol. 14, no. 4, pp. 1462–1469, 2006.

[27] V. Schmidt, K. Goyal, A. Joshi, B. Feld, L. Conell, N. Laskaris, D. Blank, J. Wilson, S. Friedler, and S. Luccioni, "CodeCarbon: Estimate and Track Carbon Emissions from Machine Learning Computing," 2021.

[28] S. Braun, H. Gamper, C. KA Reddy, and I. Tashev, "Towards efficient models for real-time deep noise suppression," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 656–660.