NEURAL NETWORK-BASED COMPRESSION FRAMEWORK FOR DOA ESTIMATION EXPLOITING DISTRIBUTED ARRAY

Saidur R. Pavel and Yimin D. Zhang

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122, USA

ABSTRACT

Distributed array consisting of multiple subarrays is attractive for high-resolution direction-of-arrival (DOA) estimation when a large-scale array is infeasible. To achieve effective distributed DOA estimation, it is required to transmit information observed at the subarrays to the fusion center, where DOA estimation is performed. For noncoherent data fusion, the covariance matrices are used for subarray fusion. To address the complexity involved with the large array size, we propose a compression framework consisting of multiple parallel encoders and a classifier. The parallel encoders at the distributed subarrays are trained to compress the respective covariance matrices. The compressed results are sent to the fusion center where the signal DOAs are estimated using a classifier based on the compressed covariance matrices.

***Index Terms*—** Direction-of-arrival estimation, distributed array, distributed sensing, data compression, neural network

1. INTRODUCTION

Distributed array consisting of multiple subarrays is attractive for high-resolution direction-of-arrival (DOA) estimation when a large-scale array is infeasible. Distributed structures are desirable in, e.g., unmanned aerial and underwater vehicle networks, where each vehicle can be equipped with a smaller number of sensors. Exploiting distributed subarrays increases system capacity, energy efficiency, robustness, and reduce multiuser interference [1–5].

To perform distributed DOA estimation, it is required that a large amount of data are transmitted from each subarray to the fusion center. For noncoherent data fusion, the covariance matrices are transmitted to the fusion center on a periodic basis. To reduce data traffic volume, we propose in this paper a data-driven compression framework based on machine learning exploiting multiple parallel encoders at the distributed subarrays and a classifier at the fusion center. We adopt an offline training strategy to train the compression framework, where the encoders and the classifier are trained jointly. This training procedure is formulated as a multilabel binary classification problem, in which the network determines whether or not each angle of the search grid contains a signal arrival. The actual signal DOAs are used as the label for training the framework by minimizing a binary cross entropy loss function. Because the classifiers and the encoder are trained jointly in this framework, the back-propagation of the loss function forces the encoders to learn, ensuring that even if the classifier is discarded, we have trained encoders. Following completion of the training, the learned network parameters are shared for online deployment. In this stage, the encoders are placed in the subarray side, and the classifier is in the fusion center. The encoders compress the covariance matrices of the received subarray signals before transmitting them to the fusion center. The classifier exploits the fused data from the fusion center to estimate the signal DOAs.

Machine learning has been extensively used in the field of image processing, speech recognition, human motion recognition, and array signal processing [6–12]. Classical machine learning techniques such as support vector regression were introduced for DOA estimation [13–15]. In [16–20], DOA estimation and source localization problems are solved based on deep learning. The proposed method differs from the existing works in several ways. In this work, we consider a distributed array consisting of multiple subarrays to perform DOA estimation. As such, network learning is focused on data compression at the distributed subarrays and fused signal detection at the fusion center. Unlike many existing schemes in which the encoders are trained for signal reconstruction, our objective is not to reconstruct the input data but rather to perform DOA estimation.

Notations: We use lower-case (upper-case) bold characters to denote vectors (matrices). In particular, $\mathbf{I}_N$ denotes the $N \times N$ identity matrix. $(\cdot)^{\mathrm{T}}$ and $(\cdot)^{\mathrm{H}}$ respectively represent the transpose and Hermitian operations of a matrix or vector. $\mathrm{triu}(\cdot)$ returns the upper triangular part of an matrix. In addition, $\mathrm{vec}(\cdot)$ vectorizes a matrix, $\mathrm{Tr}(\cdot)$ represents the trace operator, and $\mathrm{diag}(\cdot)$ forms a diagonal matrix from a vector. $\mathbb{E}[\cdot]$ stands for the statistical expectation operator. $\mathcal{R}$ and $\mathcal{I}$ respectively extract the real and imaginary parts of a complex entry. $\mathbb{C}^{M\times N}$ denotes the $M \times N$ complex space.

2. SIGNAL MODEL

Consider a distributed array consisting of K subarrays, each equipped with N antennas. D uncorrelated signals are impinging on the distributed array from directions $\boldsymbol{\theta} =$

$[\theta_1, \theta_2, \cdots \theta_D]^T$. The array received signal vector in the kth subarray, $k = 1, 2, \cdots, K$, can be modeled as

$$\begin{aligned} \boldsymbol{x}_k(t) &= \sum_{d=1}^{D} \rho_k \boldsymbol{a}_k(\theta_d) s_d(t) + \boldsymbol{n}_k(t) \\ &= \rho_k \boldsymbol{A}_k(\boldsymbol{\theta}) \boldsymbol{s}(t) + \boldsymbol{n}_k(t), \end{aligned} \tag{1}$$

where $\boldsymbol{x}_k(t)$ is the array received signal at the tth time sample and the kth subarray. $\boldsymbol{A}_k(\theta) = [\boldsymbol{a}_k(\theta_1), \boldsymbol{a}_k(\theta_2), \cdots, \boldsymbol{a}_k(\theta_D)] \in \mathbb{C}^{N \times D}$ denotes array manifold matrix whose column $\boldsymbol{a}_k(\theta_d) \in \mathbb{C}^N$ represents the steering vector of the dth user with DOA θ_d, $\boldsymbol{s}(t) = [s_1(t), s_2(t), \cdots, s_D(t)]^{\mathrm{T}} \in \mathbb{C}^D$ represents the signal waveform vector, ρ_k denotes the phase shift at the reference sensor of the kth subarray due to the physical location displacement, and $\boldsymbol{n}_k(t) \sim \mathcal{CN}(\boldsymbol{0}, \sigma_{n,k}^2 \boldsymbol{I}_N)$ represents the zero-mean additive white Gaussian noise (AWGN) vector.

The covariance matrix of the array received signal at the kth subarray can be written as

$$\begin{aligned} \boldsymbol{R}_k &= \mathbb{E}[\boldsymbol{x}_k(t) \boldsymbol{x}_k^{\mathrm{H}}(t)] = \boldsymbol{A}_k \boldsymbol{S} \boldsymbol{A}_k^{\mathrm{H}} + \sigma_{n,k}^2 \boldsymbol{I}_N \\ &= \sum_{d=1}^{D} \sigma_d^2 \boldsymbol{a}_k(\theta_d) \boldsymbol{a}_k^{\mathrm{H}}(\theta_d) + \sigma_{n,k}^2 \boldsymbol{I}_N, \end{aligned} \tag{2}$$

where $\boldsymbol{S} = \mathbb{E}(\boldsymbol{s}(t)\boldsymbol{s}^{\mathrm{H}}(t)] = \mathrm{diag}([\sigma_1^2, \sigma_2^2, \cdots, \sigma_D^2])$ is the source covariance matrix with σ_d^2 denoting the power of the dth source. The estimated covariance matrix from T sampled data for kth subarray is given as

$$\hat{\boldsymbol{R}}_k = \frac{1}{T} \sum_{t=1}^{T} \boldsymbol{x}_k(t) \boldsymbol{x}_k^{\mathrm{H}}(t). \tag{3}$$

3. NEURAL NETWORK-BASED COMPRESSION

Because the covariance matrix is a Hermitian matrix, only the upper triangular elements are considered. The upper triangular elements are vectorized before feeding into the neural network, which can be expressed for the kth subarray as

$$\bar{\boldsymbol{r}}_k = \mathrm{vec}(\mathrm{triu}(\boldsymbol{R}_k)). \tag{4}$$

The real and imaginary parts of vector $\bar{\boldsymbol{r}}_k$ is then separated and stacked as

$$\boldsymbol{r}_k = \left[(\mathcal{R}(\bar{\boldsymbol{r}}_k))^{\mathrm{T}} \; (\mathcal{I}(\bar{\boldsymbol{r}}_k))^{\mathrm{T}} \right]^{\mathrm{T}}. \tag{5}$$

The proposed network is trained offline in a simulated environment. The trained network parameters are then shared for online deployment. These two parts are respectively described in the following two subsections.

3.1. Offline Training Stage

We consider an offline training strategy as depicted in Fig. 1. In this offline training stage, K parallel encoders corresponding to K subarrays are connected to a single classifier.

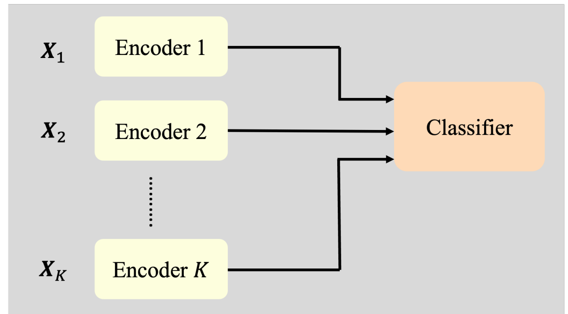


Fig. 1. Diagram for the offline training stage.

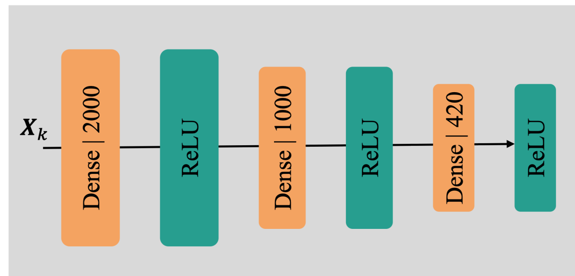


Fig. 2. Encoder structure consisting of three fully connected layers. The number in each dense layer indicates the number of neurons being used.

The encoders reduce data size in each layer and are responsible for compressing the covariance matrices at the subarrays. The resultant compressed representations obtained from the K encoders are then fed into a classifier. The entire network is then trained jointly by using the actual DOAs as the label of the network.

As depicted in Fig. 2, the encoder section employs $L = 3$ fully connected hidden layers. On the other hand, the classifier, as illustrated in Fig. 3, makes use of only one hidden layer. The number of layers is determined by balancing the nonlinear mapping between the network input and output together with the network's overfitting tendency. Each unit in a hidden layer performs logistic regression, whose output is subjected to a nonlinear activation function to introduce nonlinearity into the network.

Consider M training examples given as vectors $\boldsymbol{r}_{k,m}$, $m \in \{1, \cdots, M\}$. The input dataset of the kth encoder $\boldsymbol{X}_k$ is formed by concatenating all M vectors for the kth subarray, i.e.,

$$\boldsymbol{X}_k = [\boldsymbol{r}_{k,1}, \boldsymbol{r}_{k,2}, \cdots, \boldsymbol{r}_{k,M}]. \tag{6}$$

Considering the kth encoder, the output from the lth hidden layer is expressed as

$$\boldsymbol{Z}_k^{[l]} = \boldsymbol{W}_k^{[l]} \boldsymbol{\mathcal{A}}_k^{[l-1]} + b_k^{[l]}, \tag{7}$$

where $\boldsymbol{W}_k^{[l]}$ and $b_k^{[l]}$ respectively denote the weights and bias

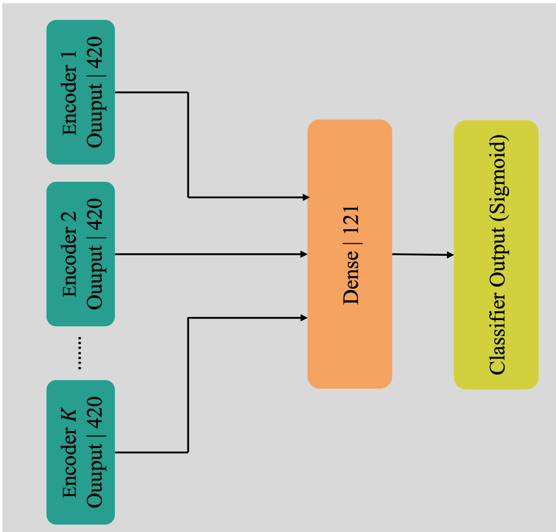


Fig. 3. Classifier structure.

corresponding to the lth hidden layer,

$$\boldsymbol{\mathcal{A}}_k^{[l]} = f\left(\boldsymbol{Z}_k^{[l]}\right), \tag{8}$$

and $f(\cdot)$ is the activation function. We use the rectified linear unit (ReLU) as the activation function in all hidden layers. Note that the input to the first hidden layer is defined as $\boldsymbol{\mathcal{A}}_k^{[0]} = \boldsymbol{X}_k$.

A dense layer with the sigmoid activation function constructs the classifier. The compressed outputs from all K encoders are used as the input of the classifier. The K parallel encoders and the classifier are jointly trained in this offline training stage. The actual signal DOAs of the training dataset are used as the label of the network, and the DOA estimation is treated as a binary classification problem. The sigmoid activation function of output nodes of the classifier produces numbers that represent the probability of the presence of a signal in each point on the search grid.

The binary cross-entropy loss function is used for the classification task. The cost function of a batch using M samples of data is expressed as

$$-\frac{1}{M}\sum_{m=1}^{M}\left[\boldsymbol{Y}^{[m]}\log\hat{\boldsymbol{Y}}^{[m]} + \left(1-\boldsymbol{Y}^{[m]}\right)\log\left(1-\hat{\boldsymbol{Y}}^{[m]}\right)\right], \tag{9}$$

where $\boldsymbol{Y}^m$ and $\hat{\boldsymbol{Y}}^m$ are, respectively, the actual and the predicted labels of the mth sample of a given batch of the training dataset.

This gradient of the cost function propagates backward through the network to jointly optimize the weights and biases of all encoders and the classifier to reduce the cost function in every epoch.

3.2. Online Deployment Stage

The previously trained encoders and the classifier are deployed in the subarrays and the fusion center, respectively. Each encoder compresses the covariance matrix of the signal vector received at the subarray, and the compressed results at all subarrays are transmitted to the fusion center. As illustrated in Fig. 3, the K compressed measurements received from the K encoders are fused to form a single measurement that is fed into the classifier. The classifier estimates the signal DOAs from the fused compressed measurements.

4. SIMULATION RESULTS

4.1. Simulation Setting

We consider a distributed array system with $K = 2$ subarrays, each equipped with $N = 50$ antennas. The antennas are arranged in a uniform linear fashion. The signals impinge into the distributed array from 5 sources, and 500 snapshots are considered.

We generate our training dataset by considering the signals impinging from directions within the range of $[-60°, 60°]$, and the input signal-to-noise ratio (SNR) is 10 dB. The entire spatial space is discretized with a $1°$ interval, rendering 121 direction grids. We consider the $K = 5$ sources with a random angular separation $\Delta\phi \in [2°, 4°, \cdots, 10°]$ between two adjacent sources, taking both equal and unequal angular separation into account.

For a specific DOA set, 10 groups of noisy snapshots are generated, rendering a total number of $310,400$ data vectors in the training dataset. 90% of them are used for training and the remainder 10% are used for validation. When we take the upper triangular elements of the 50×50 covariance matrix and vectorize the results, the total number of complex-valued elements is $1,275$. By separating the real and imaginary parts of these complex values and stacking them together, the vector size for each subarray becomes $2,550 \times 1$. These vectors are used as the inputs to jointly train the encoders and the classifier. The Adam optimizer is used to optimize the weights and biases of the network to minimize the cost function, and the learning rate is set to 0.0001. The minibatch size is set to $M = 64$ and 500 epochs are used to train the network.

4.2. Simulation Results

To evaluate the effectiveness of the performance of the proposed framework, we create a blind evaluation dataset. The parameters used in the examples of the evaluation set are both within and outside the range of parameters specified for the training data in the sense that the dataset correspond to different input SNR values and angular separations. For example, consider a test scenario with DOAs $10°, 30°, 40°, 45°, 48°$. The maximum angular separation between two adjacent sources of this test case is $20°$, which is outside the range

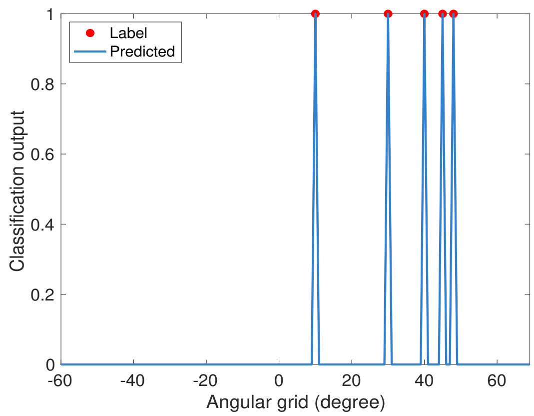
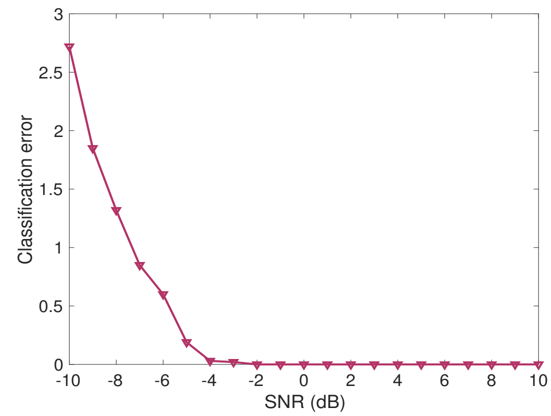


Fig. 4. DOA estimation of a test signal.



(a) Classification error versus input SNR

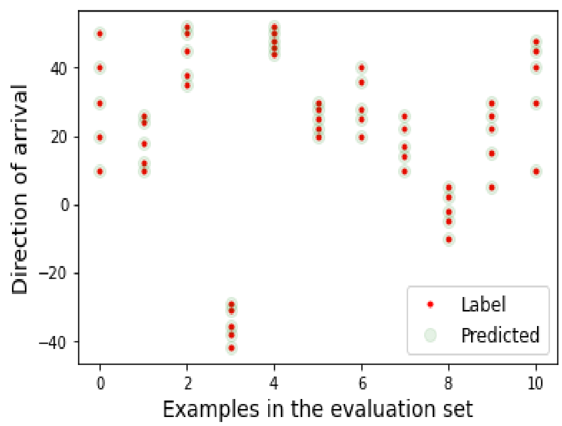
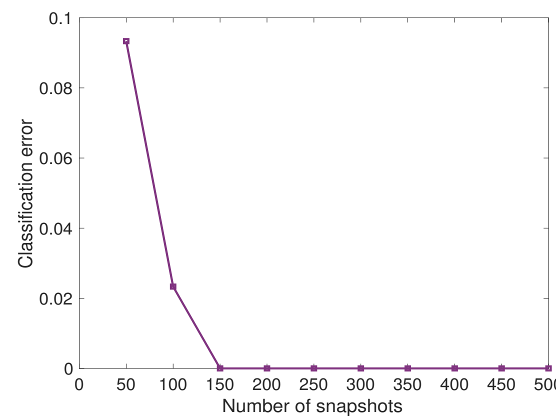


Fig. 5. Performance on the evaluation set.



(b) Classification error versus number of snapshots

Fig. 6. Classification error performance.

of the training dataset.

Fig. 4 shows the predicted DOAs corresponding to the above signal parameters, where the input SNR is 0 dB, and 500 snapshots are used. It is confirmed that the proposed framework performs correct DOA estimation for all sources even the angular separation between some of the sources is higher than the maximum angular separation of $10°$ used in the training settings.

Fig. 5 compares the predicted DOAs obtained from the proposed network and the actual label on a complete evaluation dataset. The input SNR and the number of snapshots remain unchanged. It is observed that the DOAs of all sources are correctly estimated.

Fig. 6(a) depicts the classification error performance of the proposed framework for different input SNR levels, where the number of snapshots is fixed to 500. The classification error is defined as the number of erroneously detected sources averaged over 11 different scenarios. 100 independent trials with different noise realizations are used to compute the classification error for each input SNR level. From this figure, it is observed that the proposed framework provides a low classification error when the input SNR is -4 dB or higher.

Fig. 6(b) shows the classification error performance for a varying number of snapshots, where the input SNR is 0 dB. As can be seen, the proposed framework accurately predicted the DOAs of all sources when the number of snapshots is 150 or higher.

5. CONCLUSION

In this paper, we proposed a neural network-based compression framework for transmitting compressed data from a distributed array to the fusion center to perform DOA estimation. Covariance matrices computed at each subarray are separately encoded using parallel encoders. The encoders and the classifier are jointly trained before the network parameters are shared for online deployment. The encoded data from all subarrays are then transmitted to the fusion center for data fusion and DOA estimation. A classifier network at the fusion center is used to resolve the DOAs based on the received encoded data. The proposed framework provides accurate DOA estimation performance while compressing the measurements.

6. REFERENCES

[1] M. Pesavento, A. B. Gershman, and K. M. Wong, “Direction finding in partly calibrated sensor arrays composed of multiple subarrays,” *IEEE Trans. Signal Process.*, vol. 50, no. 9, pp. 2103–2115, Sept. 2002.

[2] W. Suleiman, P. Parvazi, M. Pesavento, and A. M. Zoubir, “Non-coherent direction-of-arrival estimation using partly calibrated arrays,” *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5776–5788, Nov. 2018.

[3] A. Ahmed, S. Zhang, and Y. D. Zhang, "Multi-target motion parameter estimation exploiting collaborative UAV network," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Brighton, U.K., May 2019.

[4] P.-C. Chen and P. P. Vaidyanathan, “Distributed algorithms for array signal processing,” *IEEE Trans. Signal Process.*, vol. 69, pp. 4607–4622, 2021.

[5] M. W. T. S. Chowdhury and Y. D. Zhang, “Direction-of-arrival estimation exploiting distributed sparse arrays,” in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Oct. 2021.

[6] G. Hinton et al., “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[7] L. Zhang, F. Yang, Y. D. Zhang, and Y. J. Zhu, “Road crack detection with deep convolution neural network,” in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Phoenix, AZ, Sept. 2016.

[8] P. Kuang, T. Ma, Z. Chen, and F. Li, “Image super-resolution with densely connected convolutional networks,” *Appl. Intell.*, vol.49, pp.125–136, 2019.

[9] M. Wang, Y. D. Zhang, and G. Cui, “Human motion recognition exploiting radar with stacked recurrent neural network,” *Digital Signal Process.*, vol. 87, pp. 125–131, April 2019.

[10] H. Xian, B. Chen, T. Yang, D. Liu, “Phase enhancement model based on supervised convolutional neural network for coherent DOA estimation,” *Appl. Intell.*, vol. 50, pp. 2411–2422, 2020.

[11] Z. Cheng, H. Sun, M. Takeuchi, and J. Katto, “Deep convolutional autoencoder-based lossy image compression," in *Proc. IEEE Picture Coding Symposium (PCS)*, San Francisco, CA, June 2018.

[12] L. Theis, W. Shi, A. Cunningham, and F. Huszár, “Lossy image compression with compressive autoencoders,” arXiv preprint arXiv:1703.00395, 2017.

[13] M. Pastorino and A. Randazzo, “A smart antenna system for direction of arrival estimation based on a support vector regression,” *IEEE Trans. Antennas Propag.*, vol. 53, no. 7, pp. 2161–2168, July 2005.

[14] A. Randazzo, M. A. Abou-Khousa, M. Pastorino, and R. Zoughi, “Direction of arrival estimation based on support vector regression: Experimental validation and comparison with MUSIC,” *IEEE Antennas Wireless Propag. Lett.*, vol. 6, pp. 379–382, 2007.

[15] Y. Gao, D. Hu, Y. Chen, and Y. Ma, “Gridless 1-b DOA estimation exploiting SVM approach,” *IEEE Commun. Lett.*, vol. 21, no. 10, pp. 2210–2213, Oct. 2017.

[16] R. Takeda and K. Komatani, “Discriminative multiple sound source localization based on deep neural networks using independent location model,” in *Proc. IEEE Spoken Lang. Technol. Workshop (SLT)*,San Diego, CA, Dec. 2016, pp. 603–609.

[17] X. Xiao, S. Zhao, X. Zhong, D. L. Jones, E. S. Chng, and H. Li, “A learning-based approach to direction of arrival estimation in noisy and reverberant environments,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, South Brisbane, Queensland, Apr. 2015, pp. 2814–2818.

[18] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, “A neural network based algorithm for speaker localization in a multiroom environment,” in *Proc. IEEE Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Sept. 2016, pp. 1-–6.

[19] Z.-M. Liu, C. Zhang, and P. S. Yu, “Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections,” *IEEE Trans. Antennas Propagat.*, vol. 66, no. 12, pp. 7315–7327, 2018.

[20] S. R. Pavel, M. W. T. S. Chowdhury, Y. D. Zhang, D. Shen, and G. Chen, “Machine learning-based direction-of-arrival estimation exploiting distributed sparse arrays," in *Proc. Asilomar Conf. Signals, Systems, and Computers*, Pacific Grove, CA, Oct. 2021.