Learning Multi-Dimensional Speaker Localization: Axis Partitioning, Unbiased Label Distribution, and Data Augmentation

Linfeng Feng, *Member, IEEE*, Yijun Gong, Zhi Liu, Xiao-Lei Zhang, *Senior Member, IEEE*, and Xuelong Li, *Fellow, IEEE*

***Abstract*—Multi-dimensional speaker localization (SL) aims to estimate the two- or three-dimensional locations of speakers. A recent advancement in multi-dimensional SL is the end-to-end deep neural networks (DNNs) with ad-hoc microphone arrays. This method transforms the SL problem into a classification problem, i.e. a problem of identifying the grids where speakers are located. However, the classification formulation has two closely connected weaknesses. Firstly, this approach introduces quantization error, which needs a large number of grids to mitigate the error. However, increasing the number of grids leads to the curse of dimensionality. To address the problems, we propose an efficient multi-dimensional SL algorithm, which has the following three novel contributions. First, we decouple the high-dimensional grid partitioning into *axis partitioning*, which substantially mitigates the curse-of-dimensionality. Particularly, for the multi-speaker localization problem, we employ a separator to circumvent the permutation ambiguity of the axis partitioning in the inference stage. Second, we introduce a comprehensive *unbiased label distribution* scheme to further eliminate quantization errors. Finally, a set of data augmentation techniques are proposed, including coordinate transformation, stochastic node selection, and mixed training, to alleviate overfitting and sample imbalance problems. The proposed methods were evaluated on both simulated and real-world data, and the experimental results confirm the effectiveness.**

***Index Terms*—Multi-dimensional speaker localization, ad-hoc microphone arrays, axis partitioning, unbiased label distribution, data augmentation.**

Manuscript received 17 November 2023; revised 29 April 2024 and 17 June 2024; accepted 4 July 2024. Date of publication 25 July 2024; date of current version 13 September 2024. This work was supported in part by the National Science Foundation of China (NSFC) under Grant 62176211 and in part by the Project of the Science, Technology, and Innovation Commission of Shenzhen Municipality, China, under Grant JCYJ20210324143006016 and Grant JSGG20210802152546026. The associate editor coordinating the review of this article and approving it for publication was Dr. Romain Serizel. *(Corresponding author: Xiao-Lei Zhang.)*

Linfeng Feng and Xiao-Lei Zhang are with the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China, also with the Institute of Artificial Intelligence (TeleAI), China Telecom Corporation Ltd., Beijing 100033, China, and also with the Research and Development Institute, Northwestern Polytechnical University, Shenzhen 518063, China (e-mail: fenglinfeng@mail.nwpu.edu.cn; xiaolei.zhang@nwpu.edu.cn).

Yijun Gong is with the Xi'an Research Institute of Navigation Technology (CETC20), Xi'an 710068, China (e-mail: gongyj@mail.nwpu.edu.cn).

Zhi Liu is with the Shenzhen Huangli Techonogies Company Ltd., Shenzhen 518036, China (e-mail: liuzhi@huangliai.com).

Xuelong Li is with the Institute of Artificial Intelligence (TeleAI), China Telecom Corporation Ltd., Beijing 100033, China (e-mail: xuelong_li@ieee.org).

Digital Object Identifier 10.1109/TASLP.2024.3426309

I. Introduction

SPEAKER Localization (SL) is a technology that estimates the positions of speakers using signals captured by microphones [1]. It has many important applications, e.g. directional speech recognition [2], speech separation [3], target speaker extraction [4], speaker diarization [5], [6], etc.

A. Motivation and Challenges

The SL research originated from array signal processing techniques [7], [8], [9], [10], [11], [12], [13], [14]. Recently, researchers have shown increasingly interests in methods based on deep neural networks (DNNs) [1]. DNN-based models have proven effective in direction of arrival (DOA) estimation, showing strong performance even in noisy, reverberant environments and with multiple speakers [15], [16], [17], [18], [19], [20]. In contrast, multi-dimensional localization involves pinpointing the coordinates of speakers, which is more challenging.

DNN-based two-dimensional (2D) localization can be traced back to [21]. In their design, the output layer comprises only two neurons with the output ranging in $[0, 1]$. These neurons estimate coordinates on the horizontal plane, representing scaling factors for the room dimensions. This can be viewed as a regression problem. [22] adopts this strategy but adds an additional neuron to detect source activity. In contrast, [23], [24] subdivide the horizontal plane of a room into multiple grids, implementing classification for localization. Likewise, [25], [26] divide a room into multiple cubic blocks, enabling three-dimensional (3D) localization. In [27], the distributed array has two nodes, each acting as a sub-array. The traditional algorithm for this setup is a two-stage method: first, estimate the DOA for each node, and then conduct triangulation across the nodes. The authors proposed two end-to-end models which stack the embedding features of the nodes. It outperforms the triangulation of the separate DOAs. All the mentioned literature imposes strict constraints on the array configuration, prohibiting any changes in microphone positions or quantities from the training to the testing phase. We refer this setup as *fixed* distributed microphone arrays.

In contrast, ad-hoc microphone arrays are another type of distributed array with nodes placed randomly. When combined with deep learning, the number of nodes in these arrays can vary during both training and testing, offering flexibility in real-world

applications [28]. However, clock synchronization in distributed microphone arrays remains a significant challenge. Most existing studies assume that the nodes are synchronized. [29], [30] have made efforts to synchronize the microphone clock. Researches on SL with ad-hoc microphone arrays have been conducted in [31], [32], [33]. In [31], each node acts as a sub-array. These nodes, in conjunction with DNNs, are employed to generate DOA rays, and the intersections of these rays are determined through clustering to obtain the final speaker locations. In the configurations described by [27], [31], each node is a sub-array. The accompanying algorithms do not require time delay information interaction between nodes, which makes the asynchronicity between the nodes not an obstacle. [32] proposed an end-to-end SL model, which formulates SL as a classification problem. Unlike [23], their method introduces random alterations to the room and microphone layout and provides positional information of microphone nodes to the DNNs. This enhances the model's generalization ability to fundamentally different layouts during the test stage. A similar setting was adopted in [33] as well.

DNN-based SL models can be implemented through regression or classification. [34] notes that the classification-based SL is limited by resolution, where we need to emphasize that the word "resolution" refers to the number of classes that can be divided along a single dimension (e.g., azimuth or x-axis), and does not refer to the physical performance limits of an array. [35] found that regression on Cartesian coordinates typically yields higher accuracy, while classification shows greater robustness in challenging environments. [33] explains that the errors in the classification-based approaches consist of quantization errors and learning errors, where quantization errors are particularly severe in 2D localization. Intuitively, in a one-dimensional (1D) output space like DOA azimuth, improving resolution and increasing the number of classes are linearly related, making quantization error inversely proportional to the number of classes. However, as the number of classes increases, the model's complexity, training time, and computational costs also rise. With a large number of classes, sample imbalance can occur, where some classes have too few samples, resulting in poor learning performance for those classes. [36] proposed *Unbiased Label Distribution* (ULD) to eliminate quantization errors in classification-based azimuth DOA estimation. However, it is not suitable for multi-dimensional localization. In multi-dimensional localization, the challenge intensifies as the number of classes increases quadratically or cubically (2D or 3D) with improved resolution. This results in a marginal decrease in quantization error but a significant increase in learning error. A large number of classes can hinder a classification model's convergence during training (i.e., training loss does not decrease at all). The flexibility of ad-hoc arrays further complicates training.

Another pressing issue is severe overfitting. Ideally, a dataset should include a wide variety of microphone node locations and speaker locations. However, collecting such data is time-consuming and expensive. Therefore, SL tasks often rely on training with simulated datasets. Since simulation and real-world environments represent different domains, models trained on simulated data may not perform well on real-world data due to domain shift.

Take the example of Libri-adhoc40 [37]. This dataset comprises 40 nodes in an ad-hoc configuration, with a training set of 100 hours per node. It has proven suitable for tasks like far-field speech recognition [38], speaker verification [39], [40], and speech separation [41]. However, its training set includes data from only 9 different speaker positions, as changing speaker positions for each recording is challenging, unlike in simulations. From the perspective of SL, this represents a severe case of sample imbalance. If one were to follow the same approach as in previous tasks, pre-training on a simulation dataset first and then fine-tuning on the real-world dataset, it would likely lead to rapid overfitting. This makes fine-tuning on the real-world data unsatisfactory.

B. Goals and Contributions

To summarize the previous section, we aim to address the following critical challenges: (i) Grid partitioning encounters the curse of dimensionality, while conducting localization independently along each axis introduces permutation ambiguity in multi-speaker scenarios. (ii) The one-hot classification-based SL paradigm inevitably suffers quantization errors, even with classification accuracy reaches 100%. (iii) Severe overfitting exacerbates the difficulty of the problem. Our solutions to these challenges offer the following novel contributions:

- *We propose decoupling the output space to address the curse of dimensionality:* This approach, named *axis partitioning*, is intuitive because the coordinate axes are orthogonal. For multi-speaker localization, we suggest using an implicit separator to avoid the permutation ambiguity encountered in the axis-partitioning-based SL. The separator shifts the permutation ambiguity from the test phase to the training phase, rendering it a problem with established solutions.
- *We introduce soft labels to address quantization errors:* Quantization errors can occur both in the encoding stage of labels and the decoding stage of predicted distributions. To address this, we propose an extension for ULD to handle multi-dimensional localization.
- *We propose a set of data augmentations to mitigate overfitting:* First, we introduce *Coordinate Transformation* and *Stochastic Node Selection* to mitigate overfitting during all training stage. Secondly, to mitigate severe sample imbalance during fine-tuning on real-world datasets, we suggest using *Mixed Training*.

This paper is organized as follows. Section II introduces preliminary concepts. Sections III-A and IV explore grid and axis partitioning within the common one-hot classification paradigm. Section V offers a comprehensive description of unbiased label distributions, tailored to each partitioning method. Section VI discusses data augmentation strategies. Section VII outlines our experimental settings. SectionsVIII and IX present and analyze experimental results on simulated and real-world datasets. Finally, Section X concludes current and future work.

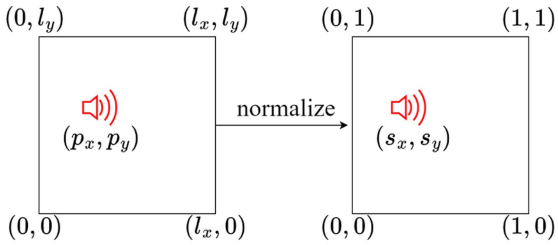


Fig. 1. Normalization process for room size.

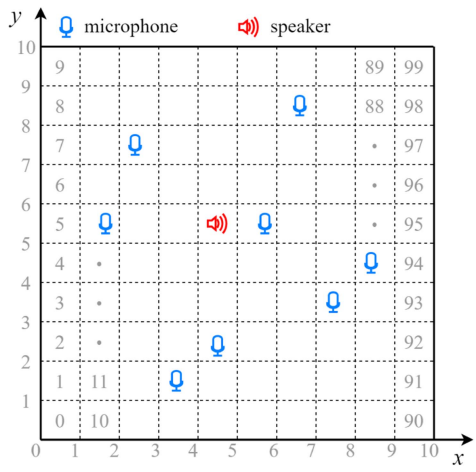


Fig. 2. The 2D plane of a normalized room is evenly divided.

II. Preliminaries

In this paper, we assume the dimensions of rooms are known. Given a room characterized by dimensions $\mathbf{l} = \{l_x, l_y, l_z\}$, we establish a Cartesian coordinate system with one corner of the room as the origin, aligning the coordinate axes with the walls' edges. Suppose a speaker is located at $\mathbf{p} = \{p_x, p_y, p_z\}$. To formulate the SL task as a learning problem, we first normalize the room to a size of $\{1, 1, 1\}$ [21]. The SL task then involves estimating the ground-truth scaling factors $\mathbf{s} = \{s_x, s_y, s_z\} = \{p_x/l_x, p_y/l_y, p_z/l_z\}$, which lie in the range $[0, 1]^3$. Subsequently, we recover the location from these predicted scaling factors. The positions of speakers and microphones are scaled using this method.

Fig. 1 illustrates the normalization process in a 2D scenario. During the inference stage, we first generate predicted scaling factors from DNNs. These scaling factors are then multiplied by the corresponding room dimensions to estimate the speaker's location.

III. Background: Grid Partitioning With One-Hot Codes

A. Grid Partitioning

First, let us consider the case of 2D grid partitioning, noting that the principle of 3D partitioning is fundamentally analogous. As illustrated in Fig. 2, we uniformly divide the normalized room into $I \times I$ grids, each labeled $\{0, 1, \ldots, I^2 - 1\}$, where $I \in \mathbb{N}$ represents the resolution. The objective of classification is to determine if the speaker is located within the i-th grid, with the grid center representing its corresponding scaling factors. The predicted 2D $\hat{\mathbf{s}}_i$ can be formalized as:

$$\hat{\mathbf{s}}_i = \{(i//I + 0.5)/I, (i\%I + 0.5)/I\} \tag{1}$$

where "//" denotes the floor division operation, and "%" denotes the modulo operation.

Similarly, we can derive the approach for 3D partitioning. We divide the room into $I \times I \times I$ grids, each labeled $\{0, 1, \ldots, I^3 - 1\}$. The predicted 3D $\hat{\mathbf{s}}_i$ can be formalized as:

$$\hat{\mathbf{s}}_i = \{(i//I//I + 0.5)/I, (i//I\%I + 0.5)/I, (i\%I^2 + 0.5)/I\} \tag{2}$$

One drawback of grid partitioning is the challenge of high dimensionality. This makes it difficult to reduce training loss, and the associated costs in training time and GPU memory can be substantial. We assert that this approach results in high-dimensional but exceedingly sparse label codings, adding unnecessary training complexities. This sparsity affects not only one-hot codes but also soft labels such as [23], [33], [36], [42], [43].

B. Multi-Speaker Localization Based on Grid Partitioning

The advantage of grid partitioning is its ease in determining the positions of multiple speakers. We introduce two methods for multi-speaker localization based on grid partitioning:

- *Multi-Label Classification (MLC):* As shown in Fig. 3, we assign a value of 1 to the grids containing speakers, and 0 otherwise [1].
- *Separator:* We decompose the multi-speaker localization task into multiple single-label classification tasks, where each speaker is assigned an independent label. To this end, we combine the grid partitioning with the implicit separator proposed in Section IV-B.

C. Quantization Errors of One-Hot Encoding and Decoding

The above grid partitioning method uses one-hot codes, which suffers quantization errors in both the training and test stages. (i) During training, quantization error arises when a real number is converted to an integer for ground-truth label. (ii) During testing, if the predicted factor is represented solely by the integer corresponding to the highest class in the predicted code, quantization error also inevitably exists.

IV. Axis Partitioning With One-Hot Codes

A. Axis Partitioning

For simplicity, our discussion will focus on partitioning the x-axis, though the same principles apply to the other axes. The x-axis is uniformly divided into I classes, labeled $\{0, 1, \ldots, I - 1\}$. The goal of classification is to determine if the speaker is in the i-th class, with the center representing its corresponding scaling factor. The predicted 1D $\hat{s}_i$ can be formalized as:

$$\hat{s}_i = (i + 0.5)/I \tag{3}$$

Taking Fig. 2 as an example, we have $I = 10$ and the speaker corresponds to $i = 4$.

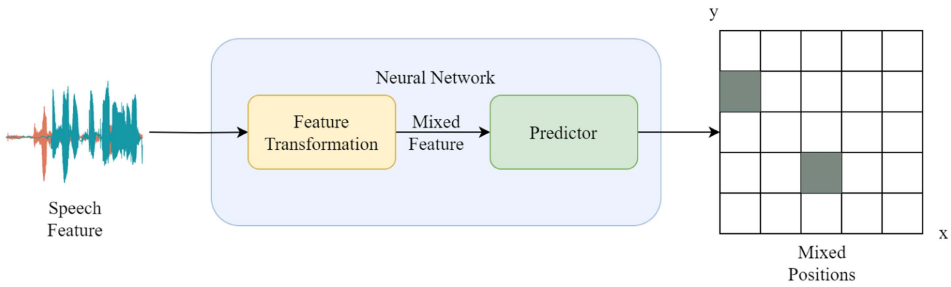


Fig. 3. The workflow of modeling multi-speaker localization as multi-label classification with gird partitioning.

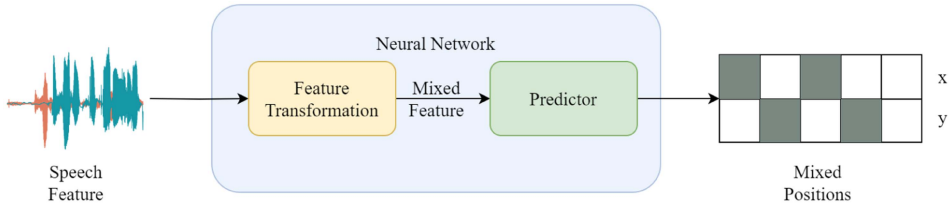


Fig. 4. The incongruence between axis partitioning and multi-label classification becomes evident in scenarios involving multiple speakers, where permutation ambiguity arise.

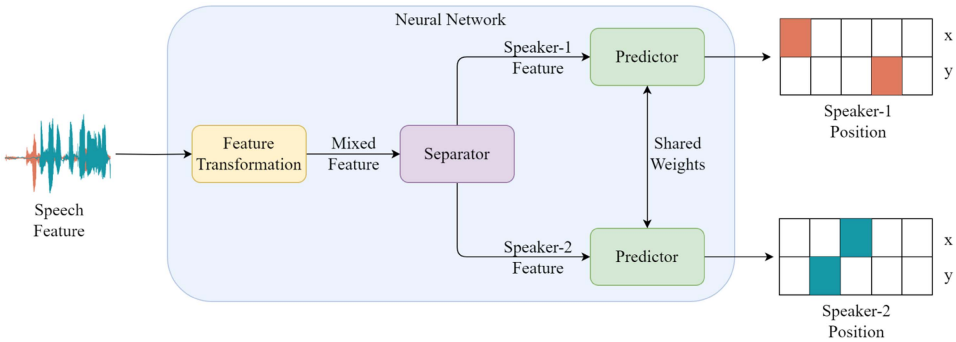


Fig. 5. After feature separation, the position of each speaker become distinctly discernible.

In the context of 3D localization, as described in Section III-A, employing grid partitioning would require I^3 output neurons of a DNN. However, with axis partitioning, only I neurons are needed to independently represent a position along each axis, resulting in a total of just $3I$ output neurons. This approach evidently enhances trainability and reduces computational costs.

B. Multi-Speaker Localization Based on Axis Partitioning

However, axis partitioning suffers from the limitation of not easily inheriting the MLC capability. Specifically, consider the two-speaker problem illustrated in Fig. 4. Using axis partitioning, one can efficiently determine the positions of the two speakers along each individual axis. These positions can be denoted as $\{s_x^1, s_x^2\}$ and $\{s_y^1, s_y^2\}$, respectively. Note that these superscripts are used solely for differentiation and do not indicate affiliation with a particular speaker. There is no definitive criterion for determining whether the positions of these two speakers should be represented as $\{(s_x^1, s_y^1), (s_x^2, s_y^2)\}$ or $\{(s_x^1, s_y^2), (s_x^2, s_y^1)\}$.

We propose using a separator, denoted as $\mathrm{Sep}(\cdot)$, to resolve the permutation ambiguity described above. Specifically, as shown in Fig. 5, we denote the embedding feature of a sound mixed from N speakers ($N > 1$) as $E \in \mathbb{R}^{T \times D}$, where T represents the number of frames, and D represents the dimension of the embedding feature at each frame. Note that E comes after the channel fusion module and therefore have no channel dimension. Given that E is a feature mixed from multiple speakers, our primary goal is to separate these individual features. Generally, achieving separation requires temporal modeling capabilities. In this paper, $\mathrm{Sep}(\cdot)$ is implemented using Bidirectional Long Short-Term Memory (BiLSTM) layers with sigmoid activation. Our approach involves the extraction of ratio masks as follows:

$$\{W_n\}_{n=1}^{N} = \mathrm{Sep}(E) \tag{4}$$

where $W_n \in [0, 1]^{T \times D}$ denotes the ratio mask of speaker n. Subsequently, we can recover the embedding feature of speaker

n by masking E with W_n:

$$\boldsymbol{e}_n = \frac{\sum_{t=1}^{T} W_n \odot E}{\sum_{t=1}^{T} W_n} \tag{5}$$

where $\boldsymbol{e}_n \in \mathbb{R}^D$ represents the embedding feature of speaker n, and $\odot$ is the element-wise product operator. This information is then processed through a predictor to obtain the predicted coding $\hat{\boldsymbol{\rho}}_n$ of speaker n:

$$\hat{\boldsymbol{\rho}}_n = \text{Pred}(\boldsymbol{e}_n) \tag{6}$$

where $\text{Pred}(\cdot)$ is composed of a linear layer with softmax activation. From $\hat{\boldsymbol{\rho}}_n$, we can easily decode the unique $(s_x^{spkr-n}, s_y^{spkr-n})$. This indicates that the permutation ambiguity during the inference phase has been resolved.

However, training the network meets the permutation ambiguity as well. Fortunately, we could overcome this problem by Permutation Invariant Training (PIT) [44]. Here we present PIT briefly as follows:

$$\mathcal{L}_{\text{PIT}} = \min_{\boldsymbol{\phi}\in\Phi} \sum_{n=1}^{N} \mathcal{L}(\hat{\boldsymbol{\rho}}_{\phi(n)}, \boldsymbol{\rho}_n), \tag{7}$$

where $\boldsymbol{\rho}_n$ is the ground-truth label coding of speaker n, $\mathcal{L}$ denotes a loss function, Φ is a set encompassing all permutations of N speakers, and $\boldsymbol{\phi}$ represents a single permutation with $\phi(n)$ representing the n-th speaker in the corresponding permutation $\boldsymbol{\phi}$.

V. Soft Label Encoding and Decoding

To eliminate the quantization errors, in this section, we first propose a novel soft label encoding and decoding strategy, named unbiased label distribution (ULD), in Section V-A. Then, we apply ULD to the grid partitioning method in Section V-B, and axis partitioning method in Section V-C. The term "unbiased" indicates that these labels are quantization-error-free. This section focuses on the scenario of single-speaker localization, while its generalization to the multi-speaker case is straightforward.

A. Unbiased Label Distribution

The ULD encoding-decoding strategy is built from a probabilistic perspective. To ensure coverage up to the boundaries, each axis is uniformly divided into $I+1$ classes, represented as $\{0, 1, \ldots, I\}$, as shown in Fig. 2.

1) Soft Encoding in the Training Stage: In the training stage, the ULD code is a vector $\boldsymbol{x} = \{x_i\}_{i=0}^{I}$. Each element of the vector falls within $[0, 1]$, and $\sum_{i=0}^{I} x_i = 1$, which can be directly interpreted as the probability of a speaker being in a particular class. Hence, we refer to the label vector as a "label distribution". Considering a ground-truth scaling factor s_x, it can be associated with the ζ-th class, where $\zeta = s_x \times I$. It is crucial to emphasize that both s_x and ζ are real numbers. The ULD vector $\boldsymbol{x}$ is defined as follows:

$$x_i = \begin{cases} 1 - \text{deci}(\zeta), & \text{if } i = \text{int}(\zeta) \\ \text{deci}(\zeta), & \text{if } i = \text{int}(\zeta) + 1 \\ 0, & \text{otherwise} \end{cases}, \quad \forall i = 0, \ldots, I \tag{8}$$

where $\text{deci}(\cdot)$ represents the extraction of the decimal portion, while $\text{int}(\cdot)$ represents the extraction of the integer portion.

Equation (8) can be intuitively understood as indicating that the speaker falls between two adjacent integer classes. For example, as shown in Fig. 2, the speaker is situated between the 4-th and 5-th classes of x-axis, and the values of the ULD can be interpreted as the probabilities of the speaker being in these two classes.

The ULD can be regarded as a smoothed and refined version of one-hot encoding [36]. Consequently, the ULD codes can directly leverage the benefits of one-hot codes for supervising the training of classification models. It can also be viewed as a form of regression. When setting I to 1, indicating binary classification using ULD, one can observe that this fundamentally aligns with the approach in [21], where regression is directly employed to estimate the scaling factors. Furthermore, increasing the number of output neurons can be seen as employing additional neurons to enhance this specific regression representation.

2) Soft Decoding in the Inference Stage: We use multiple adjacent classes for a weighted soft decoding of the speaker location $\hat{s}_x$ from $\hat{\boldsymbol{x}}$. We denote the peak class as $\hat{k} = \arg\max_i \{\hat{\boldsymbol{x}}\}_{i=0}^{I}$. The estimation $\hat{s}_x$ is a weighted summation of the adjacent classes as follows:

$$\hat{s}_x = \frac{\sum_{i=\{\hat{k}-1,\hat{k},\hat{k}+1\}} \hat{x}_i \times i/I}{\sum_{i=\{\hat{k}-1,\hat{k},\hat{k}+1\}} \hat{x}_i} \tag{9}$$

Specifically, in cases of out-of-bounds situations, i.e., when $i < 0$ or $i > I$, we set $\hat{x}_i = 0$. Note that theoretically using two adjacent classes is sufficient. However, the empirical experiments on DOA estimation, as demonstrated in [36], indicate that employing three adjacent classes with the weighted decoding almost always yields more accurate results.

B. Grid Partitioning With ULD (Grid-U)

In this section, we apply the ULD encoding-decoding strategy to the grid-partitioning-based SL. The method is denoted as *Grid-U*.

1) Soft Encoding: In the following, we present the 3D encoding method, with the 2D encoding following similar yet simpler rules.

We use the ULD soft encoding to encode each axis of a 3D location (x, y, z), which derives three tensors: $\boldsymbol{x}$ is a tensor of dimensions $(I+1) \times 1 \times 1$, $\boldsymbol{y}$ is a tensor of dimensions $1 \times (I+1) \times 1$, and $\boldsymbol{z}$ is a tensor of dimensions $1 \times 1 \times (I+1)$. A 1D label distribution can be viewed as a marginal probability distribution, whereas a multi-dimensional label distribution can be regarded as a joint probability distribution. Given that x, y, and z dimensions are mutually orthogonal, their corresponding marginal distributions are mutually independent. This property derives the joint distribution as a multiplication of the marginal distributions. Subsequently, the three tensors

are multiplied element-wisely, resulting in a 3D-ULD tensor $\boldsymbol{\rho} = \{\rho_{(i,j,k)}\}_{(i,j,k)=(0,0,0)}^{(I,I,I)}$:

$$\boldsymbol{\rho} = \boldsymbol{x} \times \boldsymbol{y} \times \boldsymbol{z} \tag{10}$$

Finally, we take $\boldsymbol{\rho}$ as the training objective of Grid-U.

2) Soft Decoding: In the test stage, suppose $\hat{\boldsymbol{\rho}}$ is an estimate of $\boldsymbol{\rho}$ made by DNN, then we can obtain the estimates of $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ by:

$$\hat{\boldsymbol{x}} = \sum_{j=0}^{I}\sum_{k=0}^{I}\hat{\boldsymbol{\rho}}, \quad \hat{\boldsymbol{y}} = \sum_{i=0}^{I}\sum_{k=0}^{I}\hat{\boldsymbol{\rho}}, \quad \hat{\boldsymbol{z}} = \sum_{i=0}^{I}\sum_{j=0}^{I}\hat{\boldsymbol{\rho}} \tag{11}$$

After obtaining $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, and $\hat{\boldsymbol{z}}$, we can employ (9) to compute the refined scaling factor $(\hat{s}_x, \hat{s}_y, \hat{s}_z)$.

C. Axis Partitioning With ULD (Axis-U)

In this section, we apply the ULD encoding-decoding strategy to the axis-partitioning-based SL. The method is denoted as *Axis-U*. The fundamental difference between Grid-U and Axis-U lies in that, the Grid-U estimates the joint distribution first and then computing the marginal distributions, while Axis-U directly estimates the marginal distributions.

1) Soft Encoding: We employ the ULD to encode each axis of a 3D location (x, y, z), resulting in three tensors denoted as $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$, each with a shape of $1 \times (I+1)$. Concatenating them along the first dimension yields $\boldsymbol{\rho} = \{\rho_{(d,i)}\}_{(d,i)=(1,0)}^{(3,I)}$:

$$\boldsymbol{\rho} = \text{Concat}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \tag{12}$$

Finally, we take $\boldsymbol{\rho}$ as the training objective of Axis-U.

2) Soft Decoding: In the test stage, we can obtain the estimates of $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ by:

$$\hat{\boldsymbol{x}} = \hat{\boldsymbol{\rho}}_1, \quad \hat{\boldsymbol{y}} = \hat{\boldsymbol{\rho}}_2, \quad \hat{\boldsymbol{z}} = \hat{\boldsymbol{\rho}}_3 \tag{13}$$

After obtaining $\hat{\boldsymbol{x}}$, $\hat{\boldsymbol{y}}$, and $\hat{\boldsymbol{z}}$, we can employ (9) to compute the refined scaling factor $(\hat{s}_x, \hat{s}_y, \hat{s}_z)$.

VI. Data Augmentation

DNN-based SL is easily overfitting, even on simulated scenarios. This section presents three data augmentation methods, including coordinate transformation, stochastic node selection, and mixed training, to alleviate the overfitting.

It is worth emphasizing that the coordinate transformation and stochastic node selection are applied to each sample in every epoch, which improves the generalization ability without increasing the computational or storage costs.

A. Coordinate Transformation

As mentioned in Section II, establishing a coordinate system requires the coordinate origin to be at one corner, and the coordinate axes to align with the corresponding edges of the walls.

The available coordinate transformations are as follows: (i) Any corner of the room can be chosen as the coordinate origin, leading to 4 possible coordinate origins in a 2D SL scenario and 8 in a 3D SL scenario. (ii) After selecting an origin, we also can have different combinations of coordinate axes. For a 2D SL scenario, there are 2 possible combinations: $\{(x, y), (y, x)\}$. For a 3D SL scenario, there are 6 possible combinations: $\{(x, y, z), (x, z, y), (y, x, z), (y, z, x), (z, x, y), (z, y, x)\}$. Eventually, for a single sample, there can be 8 possible alternatives in a 2D scene and 48 possible alternatives in a 3D scene. For instance, although both of the examples in Figs. 2 and 6 have selected the lower-left corner as the coordinate origin, the combination of the coordinate axes in Fig. 2 is (x, y), while it is (y, x) in Fig. 6.

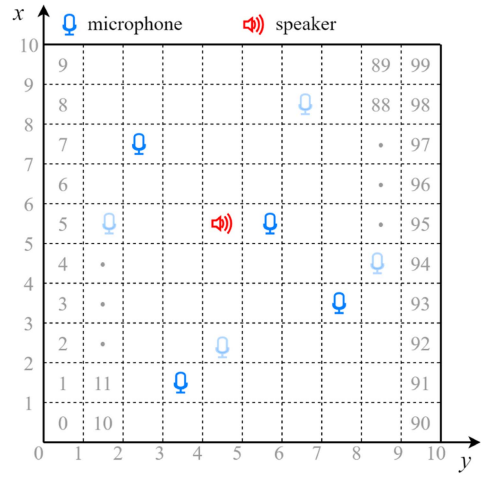


Fig. 6. A demonstration of Coordinate Transformation and Stochastic Node Selection.

B. Stochastic Node Selection

As illustrated in Fig. 6, given a large ad-hoc array, we can create many ad-hoc arrays by randomly selecting subsets of nodes of the array. If the original array has M nodes and we randomly choose M' nodes ($M \geq M'$), there are $C_M^{M'}$ different array configurations possible. This approach, named *stochastic node selection*, allows for a vast number of array configurations. Note that this approach is solely employed during the training phase.

It is easily implemented, and has the following three advantages. (i) For each utterance in a training epoch, only a subset of nodes is chosen to train the DNN, which clearly reduces computational costs and time. (ii) It allows the DNN to see as many different array configurations as possible, which provides significant performance improvement. The reason for this advantage is that changing the microphone array configuration implies a variation of spatial features which is very important for the generalization of the DNN-based SL. (iii) It can also be seen as a form of dropout regularization [45] applied to microphone nodes, which improves the generalization ability of DNN.

C. Mixed Training

Many existing speech processing techniques on ad-hoc microphone arrays follow a common workflow. It first pretrain a DNN on a simulated ad-hoc dataset, and then fine-tune the DNN

with a real-world dataset. In practice, we have observed that applying this method to SL yields catastrophic overfitting. The main reason for the failure is that the aforementioned coordinate transformations and stochastic node selection do not produce enough random number of transformations to the real-world data.

To address this issue, we propose a mixed training strategy, which uses the real-world data and simulated data together for the fine-tuning. Moreover, we propose a mixed validation strategy. It merges the simulated validation set with the real validation set into a large mixed validation set, which avoids selecting a model that is biased towards few speaker positions in the real-world validation set.

VII. Experimental Setup

A. Datasets

We have conducted experiments on both simulated and real-world datasets.

1) Simulated Data: Our simulated dataset contains 2D and 3D SL scenarios, whereas the real-world dataset focuses on the 2D SL scenario. The source speech data was from the LibriSpeech corpus [46]. We used the 'train-clean-360', 'dev-clean', and 'test-clean' subsets of Librispeech, which contain 921, 40, and 40 speakers, respectively. We adopted Pyroomacoustics [47] to generate room impulse responses. For each individual utterance, we simulated a room with a random size and extracted a 2-second portion of a clean speech. Each speech segment was then recorded using an ad-hoc microphone array consisting of 20 nodes, each equipped with a single microphone. The reverberation time T_{60} was randomly chosen between $[0.2, 1.2]$ s.

We further added noise into the reverberant speech. The additive noise was chosen from an extensive repository comprising 126 hours of diverse noise types. To control the signal-to-noise ratio (SNR), we first determine the average energy of the direct sound across all microphone channels. This average energy is then used as a reference to control the noise energy. By default, we added the same noise to each microphone. When specified, point source noise will be used. The SNR for the training set was randomly drawn from a range of $[0, 50]$ dB. The SNR for the validation and test sets was drawn randomly from a range of $[10, 20]$ dB. Note that the additive noise of the training, validation, and testing sets do not overlap.

We generates five simulated datasets, denoted as "2D-simu1", "2D-simu2", "2D-simu3", "3D-simu1", and "3D-simu2", respectively, where the terms "2D" and "3D" denote the 2D and 3D SL scenarios respectively, and the terms "simu1", "simu2", and "simu3" correspond to scenarios involving one, two, and three speakers respectively. For each dataset, the utterance number of the training, validation, and testing subsets are 18000, 1800, and 1800, respectively.

For the 2D datasets, the length and width of each room were randomly selected from a range of [4,10]m, while the height of the room was fixed at 4.2 m. The heights of the speakers were fixed at 0.95 m. The heights of the microphones was fixed at 0.9 m. To prevent the microphones or speakers too close to each other, we first partitioned a room into 16×16 grids and then placed the microphones and speakers randomly into the grids, with each grid containing at most one microphone or speaker, where the specific locations of the microphone or speaker in the grid was also random.

For the 3D datasets, we inherited most of the configurations from the 2D datasets. The difference between the 3D and 2D datasets was that the room height of the 3D datasets was randomized between [4,10]m, and the number of grids was set to $16 \times 16 \times 16$. We randomly placed microphones and speakers at all coordinates.

2) Real-World Data: The real-world data was constructed from the Libri-adhoc40 corpus [37], which incorporates 40 microphone nodes. The training set, validation set, and test set are situated at distinct speaker locations, with 9, 4, and 4 different positions, respectively.

We formulated a single-speaker 2D SL dataset from Libri-adhoc40, denoted as "2D-real1", as well as a two-speaker 2D SL dataset, denoted as "2D-real2". To generate multi-speaker datasets, we simply add the speech signals from different speakers at different locations together. For the 2D-real1 dataset, the training, validation, and test sets contain 18000, 1800, and 2468 utterances respectively. For the 2D-real2 dataset, the training, validation, and testing sets contain 18000, 1800, and 1800 utterances, respectively. Each utterance of Libri-adhoc40 was cut into 2-second segments. The geometry of the real-world room is irregular but can be approximated as a rectangular box with dimensions of approximately $9.8 \times 10.3 \times 4.2$ m. The room's T_{60} is approximately 0.9 s, with negligible presence of additive noise.

B. Parameter Settings

1) Acoustic Features: The sampling rate is 16 kHz. The window length is 512, with a hop length of 256. Short-Term Fourier Transform (STFT) features were extracted from 512 FFT with Hanning window. The real and imaginary components of the STFT were concatenated along the frequency dimension as the input of neural networks.

2) Position Codes: In this paper, the input representation of the microphones is always the same shape as the output representation. If employing one-hot encoding, each axis of a room is divided into I classes. Consequently, when using grid partitioning, the dimension of the position code, denoted as parameter C, was assigned as $C = I^2$ for the 2D SL and $C = I^3$ for the 3D SL. When using axis partitioning, these values become $C = 2I$ and $C = 3I$, respectively.

In contrast, when using ULD, each axis of a room is divided into $I + 1$ classes to cover boundaries. For the 2D SL with grid partitioning, $C = (I + 1)^2$. Similar adjustments apply in other cases.

3) Neural Networks: Two backbone neural networks were used. The first one is named the End-to-End Sound Source Localization (E2ESL) model [32]. It was applied to both the single- and multi-speaker localization using MLC. The network structure of E2ESL is detailed in Table I. Its workflow can be briefly described as follows: It contains a STFT encoder and a

TABLE I
ARCHITECTURE OF THE E2ESL / **E2ESL-SPLIT**

Module	Structure	Output size
Position encoder	Dense	512
	Dense	256
STFT encoder	Dense	1024
	Dense	512
	Dense	256
Spatial layers	Self-attention	512
	FFN	512
Fusion layer	Self-attention	512
	FFN	512
Temporal layer	Self-attention	512
	FFN	512
Output layer	Dense	1024
	Dense / **BiLSTM**	1024
	Dense	C

C denotes the length of the position code. The Feed-Forward network (FFN).

position encoder. The STFT encoder is used to transform the signals of an utterance collected by the microphone nodes into spatial-temporal embedding features of $M \times T \times D_1$, where M signifies the number of microphone nodes, T denotes the number of frames, and D_1 represents the dimension of the embedding feature at each frame. Likewise, the position encoder is used to convert the position codes of the microphone nodes into spatial-temporal embedding features characterized by dimensions $M \times T \times D_2$, where D_2 denotes the dimension of the embedding feature at each frame. In this paper, $D_1 = D_2 = 256$. Then, the two spatial-temporal embedding features were concatenated into a tensor of size $M \times T \times (D_1 + D_2)$. Then, two spatial layers perform transformations on the spatial dimension. A fusion layer combines the channels into a single one, as in [32], allowing the model to accommodate different numbers of microphone nodes between training and testing. A temporal layer processes the temporal dimension. The self-attention module in the above three kinds of layers is a 4-head attention. The high-level features produced by the above layers are then passed through fully connected layers to obtain predicted distributions.

The second backbone neural network is named E2ESL-Split. It has two differences from E2ESL. First, E2ESL-Split method is only applicable to multi-speaker localization tasks, hence necessitating the PIT for training. Second, following the approach in [43], it replaced the second-to-last layer of E2ESL by BiLSTM layers for separating the embedding features of multiple speakers.

4) Training and Evaluation Details: For all simulation experiments, we used AdamW optimizer [48] with a batch size of 16 and a maximum of 50 training epochs. The learning rate was initialized at 10^{-4}. If the validation loss did not decrease for 3 consecutive epochs, it was reduced to 10^{-5}. The best models obtained on the simulation dataset were then used for fine-tuning on the mixed dataset. The fine-tuning process involved 10 epochs with a learning rate of 10^{-5}.

TABLE II
DESCRIPTION OF THE COMPARISON METHODS, WHERE "✓" IMPLIES THAT THE OPTION IS USED

Scenario	Method	One-hot	ULD	MLC	Separator
Single	Grid	✓			
	Grid-U		✓		
	Axis	✓			
	Axis-U		✓		
Multiple	Grid	✓		✓	
	Grid-U		✓	✓	
	Grid-S	✓			✓
	Grid-U-S		✓		✓
	Axis-S	✓			✓
	Axis-U-S		✓		✓

During the training phase, for all datasets, we randomly selected 10 nodes and performed a coordinate transformation for each utterance in each epoch. During the validation and test phases, we used 20 nodes. Therefore, for the simulated datasets, all 20 nodes were selected. For the Libri-adhoc40 dataset, we randomly selected 20 nodes and performed a coordinate transformation for each utterance, due to the fixed microphone array configuration and limited speaker position options. The random seed was fixed to ensure fairness.

In order to obtain the desired predicted distributions, suitable activation and loss functions at the output layer of the backbone networks should be designed. Following [1], when performing MLC, the activation of the output layer is sigmoid, and the loss function is Binary Cross Entropy (BCE). In other cases, the activation of the output layer is softmax, and the loss function is Cross Entropy (CE). For axis partitioning, the loss of each axis is computed separately, and the total loss is the summation of them. For MLC, the mixed label code for a single sample is obtained by summing the individual label codes of each speaker. Subsequently, for each element of the mixed label code, we impose an upper limit of 1.

5) Comparison Methods: The comparison methods are summarized in Table II. From the table, we can see that the comparison methods are various combinations of the technologies referred in this paper.

6) Evaluation Metrics: Suppose a dataset has N^{total} test speakers. Then, a metric is the classification accuracy (ACC):

$$\text{ACC}(\%) = \frac{N^{\text{acc}}}{N^{\text{total}}} \times 100 \tag{14}$$

where N^{acc} represents the number of speakers whose predicted location is correctly classified into the ground-truth class. When applying ACC to evaluate axis partitioning methods, the ACC for each axis was first evaluated independently. Then, the ACC values across all axes are averaged to determine the overall ACC. A lower ACC suggests that the model is more challenging to train, and if ACC becomes too low, it indicates non-convergence. Therefore, the aforementioned definition of ACC is mainly used to compare the convergence difficulty of classification models,

TABLE III
RESULTS ON THE SINGLE-SPEAKER TWO-DIMENSIONAL SIMULATED DATA "2D-SIMU1"

	I	1	2	4	8	16	32	64
QE	MAE	2.705	1.340	0.687	0.340	0.174	0.086	0.043
Grid	ACC	100.00	90.44	76.82	59.14	31.52	/	/
	MAE	2.705	1.391	0.794	0.513	**0.405**	/	/
Grid-U	ACC	95.28	89.33	78.77	66.70	48.64	10.84	/
	MAE	0.338	0.307	0.273	**0.234**	0.236	0.380	/
Axis	ACC	100.00	95.14	88.78	80.42	68.47	50.42	24.17
	MAE	2.705	1.386	0.774	0.456	0.309	**0.238**	0.275
Axis-U	ACC	98.03	93.83	88.83	80.96	71.46	51.17	30.43
	MAE	0.348	0.326	0.270	0.248	**0.223**	0.240	0.242

TABLE IV
RESULTS ON THE SINGLE-SPEAKER THREE-DIMENSIONAL SIMULATED DATA "3D-SIMU1"

	I	1	2	4	8	16	32	48
QE	MAE	3.413	1.708	0.858	0.428	0.214	0.106	0.071
Grid	ACC	100.00	62.67	42.00	/	/	/	/
	MAE	3.413	1.996	**1.192**	/	/	/	/
Grid-U	ACC	92.94	79.17	63.56	31.00	/	/	/
	MAE	0.469	0.465	**0.414**	0.536	/	/	/
Axis	ACC	100.00	89.92	80.50	67.20	70.95	41.02	23.96
	MAE	3.413	1.826	1.074	0.654	0.444	**0.397**	0.443
Axis-U	ACC	97.28	93.02	85.33	74.98	59.74	41.19	27.59
	MAE	0.500	0.461	0.413	0.404	0.387	**0.385**	0.409

and is therefore not suitable for comparing the localization accuracy of models.

Another evaluation metric for SL is the mean absolute error (MAE) between the predicted location and the ground-truth location in the Euclidean space:

$$\mathrm{MAE}(m) = \frac{1}{N^{\text{total}}} \sum_{n=1}^{N^{\text{total}}} ||\mathbf{p}_n - \hat{\mathbf{p}}_n||_2 \tag{15}$$

where $||\cdot||_2$ denotes the Euclidean norm. When there are multiple speakers, there are many possible combinations between the multiple predicted positions and ground-truth positions, each of which yields a MAE. We selected the combination that results in the minimum MAE as the best one.

We use 'QE' to denote quantization error, which is the MAE incurred even when achieving 100% classification accuracy using the one-hot paradigm. The symbol '/' indicates that the model training fails to converge.

VIII. RESULTS ON SIMULATED DATA

A. Results on Single-Speaker Localization

The main purposes of the experiments on the single-speaker localization are to show (i) the effectiveness of the axis-partitioning-based methods in dealing with the curse of dimensionality, and (ii) the effectiveness of ULD in overcoming the quantization errors caused by the one-hot paradigm.

1) Two-Dimensional Localization: In Table III, we can observe that the quantization error is clearly inversely proportional to I, which aligns with our theoretical expectations. This serves as the lower bound for the methods of both 'Grid' and 'Axis'. When I is relatively small, DNNs can easily achieve high ACC, and the MAE for both 'Grid' and 'Axis' closely approaches QE. However, as I gradually increases, the ACC decreases, and learning errors gradually dominate the total error over quantization errors. Given the same I, axis partitioning generally achieves higher ACC than grid partitioning because the number of classes is much smaller. When $I = 16$, the ACC for 'Grid' is already quite low; when I is further increased to 32 which corresponds to 1024 classes, its training no longer converges. On the other side, even when $I = 64$, 'Axis' with 128 classes can still converge to achieve decent results. This indicates that 'Axis' exhibits better trainability than 'Grid'. After integrating with ULD, 'Grid-U' converges at $I = 32$. This phenomenon may be attributed to ULD's ability to eliminate quantization errors, thereby making the input position information accurate. The best MAE was achieved by 'Axis-U', which was 44.94% relatively lower than the original 'Grid'.

2) Three-Dimensional Localization: A more challenging scenario is the 3D SL. The rooms are larger without height restrictions, which significantly increases the occurrence likelihood of far-field scenarios. The sound sources are not confined to a plane, leading to a larger range of error fluctuations.

Table IV show that the models of the grid-partitioning-based methods are difficult to be trained successfully due to the curse of dimensionality. Even if they achieve convergence, they yield very low accuracy; while the axis-partitioning-based methods do not have such difficulty, which demonstrates clearly the advantages over the grid-partitioning-based methods. Furthermore, when ULD is applied, 'Axis-U' reduces MAE by 67.70% over 'Grid'.

B. Results on Multi-Speaker Localization

The main purposes of the experiments on the multi-speaker localization are mainly to show the advantage of separator-based SL over MLC-based SL.

1) Two-Dimensional Localization: Table V lists the comparison result on the 2D two-speaker localization. From the table, we see that the proposed separator-based methods outperform the MLC-based methods. For the separator-based methods, it is apparent that 'Grid-S' and 'Grid-U-S' cannot be trained successfully when I becomes large, while 'Axis-S' and 'Axis-U-S' still maintain their trainability. When $I = 1$, 'Axis-U-S' essentially performs like a regression [21]. Moreover, it exhibits lower performance than the typical classification 'Grid-S' when $I = 8$. A similar phenomenon has been observed in [35] where the authors claimed that classification is more robust than regression for SL. From $I = 1$ to $I = 16$, 'Axis-U-S' achieves large improvement, indicating that ULD can inherit the robustness of the classification-based SL. To summarize, with the support of ULD, 'Axis-U-S' reduces MAE by 41.70% over 'Grid-S', and 53.32% over the regression-based SL.

TABLE V
RESULTS ON THE TWO-SPEAKER TWO-DIMENSIONAL SIMULATED DATA "2D-SIMU2"

	I	1	2	4	8	16	32	64
QE	MAE	2.695	1.346	0.664	0.339	0.170	0.085	0.042
Grid	ACC	100.00	75.25	51.44	34.78	/	/	/
	MAE	2.695	1.902	**1.810**	1.847	/	/	/
Grid-U	ACC	76.78	65.39	51.11	36.06	/	/	/
	MAE	2.049	2.052	1.903	**1.853**	/	/	/
Grid-S	ACC	100.00	88.58	72.56	53.28	/	/	/
	MAE	2.695	1.434	0.839	**0.578**	/	/	/
Grid-U-S	ACC	89.14	80.33	71.28	58.61	35.06	/	/
	MAE	0.694	0.561	0.413	**0.342**	0.376	/	/
Axis-S	ACC	100.00	89.36	79.30	74.81	60.11	39.59	19.65
	MAE	2.695	1.477	0.904	0.515	0.397	**0.383**	0.452
Axis-U-S	ACC	93.71	88.60	83.28	72.03	61.97	40.44	23.03
	MAE	0.722	0.571	0.438	0.423	**0.337**	0.378	0.400

TABLE VI
RESULTS ON THE THREE-SPEAKER TWO-DIMENSIONAL SIMULATED DATA "2D-SIMU3"

	I	1	2	4	8	16	32
QE	MAE	3.590	1.793	0.822	0.366	0.168	0.084
Grid	ACC	100.00	85.82	48.62	35.42	/	/
	MAE	3.590	2.417	**2.295**	2.341	/	/
Grid-U	ACC	85.80	58.44	48.30	35.26	/	/
	MAE	2.664	2.554	2.412	**2.310**	/	/
Grid-S	ACC	100.00	85.48	68.11	44.31	/	/
	MAE	3.590	1.517	1.005	**0.890**	/	/
Grid-U-S	ACC	84.70	74.76	63.35	49.91	/	/
	MAE	0.909	0.774	0.635	**0.630**	/	/
Axis-S	ACC	100.00	90.69	80.25	63.82	39.52	25.96
	MAE	3.590	1.548	1.042	0.853	1.069	**0.849**
Axis-U-S	ACC	91.19	85.67	77.36	66.47	48.63	29.59
	MAE	0.933	0.809	0.694	**0.545**	0.666	0.693

Table VI lists the comparison result on the 2D three-speaker localization problem. From the result, we can see that increasing extra speakers affects the localization performance significantly. However, the proposed technique still achieves the best performance. For example, the MAE of 'Axis-U-S' is 38.76% lower than that of 'Grid-S'.

2) Three-Dimensional Localization: Table VII lists the comparison results on the 3D SL scenario. From the table, it is clear that most of our observations are similar to those in the other scenarios. Specifically, the axis-partitioning-based methods consistently outperforms the grid-partitioning-based methods. ULD outperforms the one-hot strategy. The separator-based methods outperform the MLC-based methods. 'Axis-U-S' reduces MAE by 40.03% over 'Grid-S'. Although 'Grid-U-S' achieves the best performance, we insistently recommend 'Axis-U-S'. The reason is as follows: as illustrated in Table VIII, the model size of grid partitioning grows dramatically with the increase of I, while the model size of axis partitioning grows linearly with I which facilitates its practical use.

TABLE VII
RESULTS ON THE TWO-SPEAKER THREE-DIMENSIONAL SIMULATED DATA "3D-SIMU2"

	I	1	2	4	8	16	32	48
QE	MAE	3.398	1.710	0.849	0.428	0.214	0.106	0.071
Grid	ACC	100.00	53.03	36.50	/	/	/	/
	MAE	3.398	3.098	**2.074**	/	/	/	/
Grid-U	ACC	85.50	67.56	50.44	/	/	/	/
	MAE	2.457	1.907	**1.571**	/	/	/	/
Grid-S	ACC	100.00	52.67	35.11	/	/	/	/
	MAE	3.398	2.207	**1.399**	/	/	/	/
Grid-U-S	ACC	77.50	59.61	45.81	24.92	/	/	/
	MAE	1.207	0.956	0.816	**0.739**	/	/	/
Axis-S	ACC	100.00	85.31	73.17	58.04	37.97	23.04	/
	MAE	3.398	2.038	1.292	0.884	**0.883**	0.912	/
Axis-U-S	ACC	92.50	80.31	73.97	59.27	39.27	20.40	15.81
	MAE	1.191	1.299	0.903	**0.839**	0.854	1.045	0.959

TABLE VIII
THE INFLUENCE OF I ON THE NUMBER OF MODEL PARAMETERS (IN MILLIONS), WHERE THE MODELS WERE TRAINED ON THE SIMULATED DATA "3D-SIMU2"

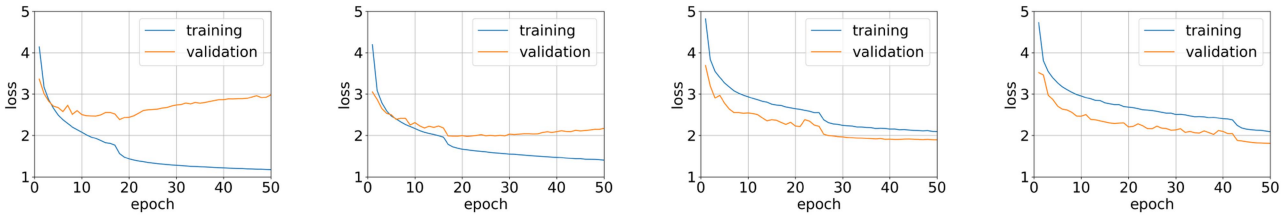
	1	2	4	8	16	32	48
Grid	8.55	8.56	8.65	9.34	14.84	58.91	178.53
Grid-U	8.56	8.59	8.74	9.67	16.10	63.78	189.38
Grid-S	13.80	13.81	13.90	14.58	20.09	64.16	183.78
Grid-U-S	13.81	13.84	13.99	14.92	21.35	69.03	194.62
Axis-S	13.80	13.81	13.82	13.83	13.87	13.94	14.02
Axis-U-S	13.81	13.81	13.82	13.84	13.88	13.95	14.02

C. Empirical Study on Data Augmentation

Fig. 7 shows the effects of the components of the novel data augmentation on performance. Specifically, Fig. 7(a) illustrates that without employing data augmentation, the training loss decreases rapidly, leading to early overfitting. In contrast, Fig. 7(b) incorporates coordinate transformation. It effectively augmented the dataset, which significantly degrades the validation loss. An interesting phenomenon in Fig. 7(c) is that the validation loss is even lower than the training loss. This is because that the training data which consists of 10 ad-hoc nodes per utterance is more challenging than the validation data which contains 20 ad-hoc nodes. As shown in Fig. 7(d), integrating the coordinate transformation and stochastic node selection together achieves a lower validation loss. Table IX lists the effects of the proposed CA data augmentation and its components on the test set. From the table, we see that the components, i.e. CT and SNS, are effective, while CA outperforms the method without augmentation by a relative MAE reduction of 33.4%.

D. Empirical Study on Different Noises

We studied the effectiveness of various methods in different noisy environments. First, we duplicated '2D-simu1' and added



(a) No data augmentation (b) Coordinate transformation (c) Stochastic node selection (d) Composite augmentation

Fig. 7. Ablation study of the data augmentation methods on the 2D-simu1 dataset, where 'Axis-U' with $I = 16$ was used.

TABLE IX
THE RESULTS OF DATA AUGMENTATIONS ON THE 2D-SIMU1 DATASET, WHERE 'AXIS-U' WITH $I = 16$ WAS USED

	No data augmentation	CT	SNS	CA
ACC	61.53	68.40	67.95	71.46
MAE	0.333	0.259	0.249	0.223

CT is short for coordinate transformation, SNS short for stochastic sode selection, and CA short for composite augmentation.

TABLE X
RESULTS ON DIFFERENT SNR AND NOISE TYPES, WHERE EACH METHOD USING THE OPTIMAL I

Noise type	Diffuse noise			Point source noise		
SNR	0 dB	10 dB	20 dB	0 dB	10 dB	20 dB
Grid	0.660	0.580	0.568	0.908	0.617	0.559
Grid-U	0.201	0.157	0.150	0.262	0.173	0.151
Axis	0.226	0.182	0.172	0.317	0.197	0.174
Axis-U	**0.170**	**0.133**	**0.126**	**0.244**	**0.148**	**0.128**

diffuse noise to one copy and point source noise to the other. The training sets from these two noise environments were combined into a large training set, and the validation sets were similarly merged. These sets were used to train models that generalize across different noise types. Each utterance in the training and validation sets had an SNR randomly chosen from $[0, 20]$ dB. The test set's SNR was controlled at $\{0, 10, 20\}$ dB.

Table X presents the comparison results on diffuse noise and point source noise respectively. It is evident that point source noise is more challenging than diffuse noise. The inclusion of point source noise slow the convergence of the 'Grid', resulting in poor performance. However, the proposed axis-partitioning and ULD methods demonstrate their relative effectiveness respectively. The combined 'Axis-U' model performs best, achieving a small error of 0.244 m even in a 0 dB point source noise environment.

IX. RESULTS ON REAL-WORLD DATA

A. Main Results

Tables XI and XII show the performance of the comparison methods on the single- and two-speaker real-world data respectively. We see similar experimental phenomena as those on the simulated data. Specifically, the performance of the methods based on one-hot paradigm is lower-bounded by QE. While

TABLE XI
RESULTS ON THE SINGLE-SPEAKER TWO-DIMENSIONAL REAL-WORLD DATA '2D-REAL1'

	I	1	2	4	8	16	32	64
QE	MAE	4.306	1.804	0.716	0.650	0.290	0.116	0.042
Grid	ACC	100.00	90.64	87.48	75.61	62.44	/	/
	MAE	4.306	1.903	0.784	0.734	**0.394**	/	/
Grid-U	ACC	99.92	85.33	54.86	75.97	64.59	65.96	/
	MAE	0.445	0.347	0.295	**0.209**	0.253	0.231	/
Axis	ACC	100.00	91.48	97.18	93.46	87.00	82.77	82.43
	MAE	4.306	2.081	1.396	0.411	0.285	**0.225**	0.230
Axis-U	ACC	99.90	92.48	79.07	86.77	83.39	75.55	65.46
	MAE	0.435	0.382	0.300	0.229	0.201	0.244	**0.187**

TABLE XII
RESULTS ON THE TWO-SPEAKER TWO-DIMENSIONAL REAL-WORLD DATA '2D-REAL2'

	I	1	2	4	8	16	32	64
QE	MAE	4.318	1.787	0.723	0.653	0.288	0.115	0.042
Grid	ACC	100.00	81.25	47.14	8.44	/	/	/
	MAE	4.318	2.810	1.747	**1.645**	/	/	/
Grid-U	ACC	99.64	62.17	42.78	16.86	/	/	/
	MAE	1.967	1.718	**1.358**	1.479	/	/	/
Grid-S	ACC	100.00	86.06	66.19	54.67	/	/	/
	MAE	4.318	1.997	1.168	**1.079**	/	/	/
Grid-U-S	ACC	90.58	75.22	19.53	60.81	58.28	/	/
	MAE	1.169	1.035	0.788	0.558	**0.501**	/	/
Axis-S	ACC	100.00	76.69	77.20	64.17	72.36	63.56	67.56
	MAE	4.318	2.544	1.662	0.718	0.702	**0.668**	0.739
Axis-U-S	ACC	94.08	85.58	50.35	79.79	73.17	66.40	63.50
	MAE	1.132	1.025	0.904	0.663	**0.518**	0.634	0.718

the ULD-based methods significantly breaks such limitation when I is small. Axis-partitioning-based methods continue to outperform their grid partitioning counterparts in most cases. We also found that the results in Table XI generally are better than those in Table III, even though Libri-adhoc40 has a large room with high reverberation. This phenomenon may be caused by that the simulated dataset includes environmental noise, whereas Libri-adhoc40 includes little additive noise. In Table XII, although 'Grid-U-S' slightly outperforms 'Axis-U-S', we still recommend 'Axis-U-S', due to its low computational complexity. To summarize, 'Axis-U' outperforms 'Grid' by a relative MAE reduction of 52.54% on 2D-real1, and 'Axis-U-S'

TABLE XIII
THE RESULTS OF DATA AUGMENTATIONS ON THE REAL DATASET, WHERE 'AXIS-U' WITH $I = 16$ WAS USED AS THE SL MODEL

	Pre-train	Real fine-tuning	Mixed fine-tuning
ACC	44.06	45.19	83.39
MAE	0.648	0.634	0.201

'Real fine-tuning' refers to the method of fine-tuning the pre-trained model using real datasets, and so forth.

outperforms 'Grid-S' by a relative MAE reduction of 51.99% on 2D-real1.

B. Empirical Study on Data Augmentation

Table XIII shows the effect of the mixed-training strategy in the real-world scenario. From the table, we can see that directly applying the pre-trained model to the real dataset does not perform well. Using the real-world data to fine tune the pre-trained model still yields unsatisfactory results. In contrast, fine-tuning the pre-trained model with a combination of both the simulated and real-world datasets alleviates the issue of sample imbalance, resulting in a relative MAE reduction of 68.95% over the pre-trained model.

X. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed several algorithms for the multi-dimensional speaker localization. Notably, we introduced axis partitioning to address the curse of dimensionality. Additionally, we suggested using a separator to resolve the permutation ambiguity problem inherent in axis partitioning when handling multi-speaker localization tasks. Furthermore, we present a comprehensive strategy of unbiased label distribution to mitigate quantization errors. At last, we proposed several data augmentation methods, including coordinate transformation, stochastic node selection, and mixed training, to mitigate the overfitting. We have conducted extensive experiments on both simulated and real-world data, and the results show the effectiveness of the proposed algorithms.

The research on end-to-end SL with ad-hoc arrays is still at the beginning, with numerous open problems. We highlight several pressing topics as follows: (i) Developing methods to handle various types of sound sources, including moving sources and interference beyond speakers. (ii) Investigating whether enhancing the separator, potentially through more sophisticated temporal processing and fusion techniques, leads to performance gains and cost savings. (iii) Minimizing reliance on a priori information about room dimensions and microphone positions, and studying input representations for microphone positions. (iv) Combining neural networks with array processing techniques, such as designing an effective node selection algorithm for ad-hoc microphone arrays. Addressing these challenges would markedly advance the field, not only enhancing existing systems but also opening new avenues for research and applications.

REFERENCES

[1] P.-A. Grumiaux, S. Kitić, L. Girin, and A. Guérin, "A survey of sound source localization with deep learning methods," *J. Acoustical Soc. Amer.*, vol. 152, no. 1, pp. 107–151, 2022.

[2] A. S. Subramanian et al., "Directional ASR: A new paradigm for E2E multi-speaker speech recognition with source localization," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 8433–8437.

[3] Z.-Q. Wang and D. Wang, "Localization based sequential grouping for continuous speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 281–285.

[4] M. Ge, C. Xu, L. Wang, E. S. Chng, J. Dang, and H. Li, "L-SpEx: Localized target speaker extraction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 7287–7291.

[5] H. Taherian and D. Wang, "Multi-channel conversational speaker separation via neural diarization," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 32, pp. 2467–2476, 2024.

[6] T. Gburrek, J. Schmalenstroeer, and R. Haeb-Umbach, "Spatial diarization for meeting transcription with ad-hoc acoustic sensor networks," in *Proc. IEEE 57th Asilomar Conf. Signals, Syst., Comput.*, 2023, pp. 1399–1403.

[7] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Trans. Acoustics, Speech, Signal Process.*, vol. ASSP-24, no. 4, pp. 320–327, Aug. 1976.

[8] R. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. Antennas Propag.*, vol. TAP-34, no. 3, pp. 276–280, Mar. 1986.

[9] J. Smith and J. Abel, "The spherical interpolation method of source localization," *IEEE J. Ocean. Eng.*, vol. 12, no. 1, pp. 246–252, Jan. 1987.

[10] P. Stoica and K. C. Sharman, "Maximum likelihood methods for direction-of-arrival estimation," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 38, no. 7, pp. 1132–1143, Jul. 1990.

[11] Y. T. Chan and K. C. Ho, "A simple and efficient estimator for hyperbolic location," *IEEE Trans. Signal Process.*, vol. 42, no. 8, pp. 1905–1915, Aug. 1994.

[12] J. H. DiBiase, "A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays," Brown University, 2000.

[13] J. Scheuing and B. Yang, "Correlation-based TDOA-estimation for multiple sources in reverberant environments," *Speech Audio Process. Adverse Environments*, pp. 381–416, 2008.

[14] R. Amiri, F. Behnia, and A. Noroozi, "An efficient estimator for TDOA-based source localization with minimum number of sensors," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2499–2502, Dec. 2018.

[15] S. Chakrabarty and E. A. P. Habets, "Multi-speaker DOA estimation using deep convolutional networks trained with noise signals," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 1, pp. 8–21, Mar. 2019.

[16] W. Zhang, Y. Zhou, and Y. Qian, "Robust DOA estimation based on convolutional neural network and time-frequency masking," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 2703–2707.

[17] Y. Fu et al., "Iterative sound source localization for unknown number of sources," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2022, pp. 896–900.

[18] Q. Hu, N. Ma, and G. J. Brown, "Robust binaural sound localisation with temporal attention," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.

[19] U. Kowalk, S. Doclo, and J. Bitzer, "Geometry-aware DOA estimation using a deep neural network with mixed-data input features," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.

[20] P. Cooreman, A. Bohlender, and N. Madhu, "CRNN-based multi-DOA estimator: Comparing classification and regression," in *Proc. IEEE Speech Commun.; 15th ITG Conf.*, 2023, pp. 156–160.

[21] F. Vesperini, P. Vecchiotti, E. Principi, S. Squartini, and F. Piazza, "A neural network based algorithm for speaker localization in a multi-room environment," in *Proc. IEEE 26th Int. Workshop Mach. Learn. Signal Process.*, 2016, pp. 1–6.

[22] P. Vecchiotti, G. Pepe, E. Principi, and S. Squartini, "Detection of activity and position of speakers by using deep neural networks and acoustic data augmentation," *Expert Syst. with Appl.*, vol. 134, pp. 53–65, 2019.

[23] G. Le Moing et al., "Learning multiple sound source 2D localization," in *Proc. IEEE 21st Int. Workshop Multimedia Signal Process.*, 2019, pp. 1–6.

[24] D. Zhang, J. Chen, J. Bai, M. S. Ayub, M. Wang, and Q. Yan, "Multiple sound sources localization using sub-band spatial features and attention mechanism," Available at SSRN 4618444.

[25] Y. Sun, J. Chen, C. Yuen, and S. Rahardja, "Indoor sound source localization with probabilistic neural network," *IEEE Trans. Ind. Electron.*, vol. 65, no. 8, pp. 6403–6413, Aug. 2018.

[26] J. Yan, W. Zhao, Y. I. Wu, and Y. Zhou, "Indoor sound source localization under reverberation by extracting the features of sample covariance," *Appl. Acoust.*, vol. 210, 2023, Art. no. 109453.
[27] S. Kindt, A. Bohlender, and N. Madhu, "2D acoustic source localisation using decentralised deep neural networks on distributed microphone arrays," in *Proc. IEEE Speech Commun.; 14th ITG Conf.*, 2021, pp. 1–5.
[28] X.-L. Zhang, "Deep ad-hoc beamforming," *Comput. Speech Lang.*, vol. 68, 2021, Art. no. 101201.
[29] T. Gburrek, J. Schmalenstroeer, and R. Haeb-Umbach, "On synchronization of wireless acoustic sensor networks in the presence of time-varying sampling rate offsets and speaker changes," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2022, pp. 916–920.
[30] A. Chinaev, N. Knaepper, and G. Enzner, "Long-term synchronization of wireless acoustic sensor networks with nonpersistent acoustic activity using coherence state," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.
[31] S. Liu, Y. Gong, and X.-L. Zhang, "Deep learning based two-dimensional speaker localization with large ad-hoc microphone arrays," 2022, *arXiv:2210.10265*.
[32] Y. Gong, S. Liu, and X.-L. Zhang, "End-to-end two-dimensional sound source localization with ad-hoc microphone arrays," in *Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2022, pp. 1944–1949.
[33] L. Feng, Y. Gong, and X.-L. Zhang, "Soft label coding for end-to-end sound source localization with ad-hoc microphone arrays," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2023, pp. 1–5.
[34] Z.-Q. Wang, X. Zhang, and D. Wang, "Robust TDOA estimation based on time-frequency masking and deep neural networks," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 322–326.
[35] L. Perotin, A. Défossez, E. Vincent, R. Serizel, and A. Guérin, "Regression versus classification for neural network based audio source localization," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, 2019, pp. 343–347.
[36] L. Feng, X.-L. Zhang, and X. Li, "Eliminating quantization errors in classification-based sound source localization," *Neural Netw.*, vol. 106679, pp. 1-13, 2024.
[37] S. Guan et al., "Libri-adhoc40: A dataset collected from synchronized ad-hoc microphone arrays," in *Proc. IEEE Asia-Pacific Signal Inf. Process. Assoc. Annu. Summit Conf.*, 2021, pp. 1116–1120.
[38] J. Chen and X.-L. Zhang, "Scaling sparsemax based channel selection for speech recognition with ad-hoc microphone arrays," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 291–295.
[39] C. Liang, Y. Chen, J. Yao, and X.-L. Zhang, "Multi-channel far-field speaker verification with large-scale ad-hoc microphone arrays," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2022, pp. 3679–3683.
[40] Y. Chen, C. Liang, and X.-L. Zhang, "Spatial-temporal graph based multi-channel speaker verification with ad-hoc microphone arrays," 2023, *arXiv:2307.01386*.
[41] Z. Yang, S. Guan, and X.-L. Zhang, "Deep ad-hoc beamforming based on speaker extraction for target-dependent speech separation," *Speech Commun.*, vol. 140, pp. 87–97, 2022.
[42] W. He, P. Motlicek, and J.-M. Odobez, "Deep neural networks for multiple speaker detection and localization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 74–79.
[43] A. S. Subramanian, C. Weng, S. Watanabe, M. Yu, and D. Yu, "Deep learning based multi-source localization with source splitting and its effectiveness in multi-talker speech recognition," *Comput. Speech Lang.*, vol. 75, 2022, Art. no. 101360.
[44] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2017, pp. 241–245.
[45] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.
[46] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2015, pp. 5206–5210.
[47] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process*, 2018, pp. 351–355.
[48] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2019, pp. 1–19.

Linfeng Feng (Member, IEEE) is currently working toward the Ph.D. degree in information and communication engineering with the Northwestern Polytechnical University, Xi'an, China. He is an Algorithm Intern with the Institute of Artificial Intelligence, China Telecom, Beijing, China. His research interests include spatial audio, speech enhancement, and multimodal learning.

Yijun Gong received the master's degree in electronic information from Northwestern Polytechnical University, Xi'an, China, in 2023. His research focuses on acoustic signal processing.

Zhi Liu received the master's degree in information and communication engineering from Tsinghua University, Beijing, China, in 2004. He is currently the General Manager of Oleap Technology International Limited, Shenzhen, China. His research interests include speech enhancement, speech separation, and artificial intelligence.

Xiao-Lei Zhang (Senior Member, IEEE) received the Ph.D. degree in information and communication engineering from Tsinghua University, Beijing, China, in 2012. He was a Postdoctoral Researcher with the Perception and Neurodynamics Laboratory, The Ohio State University, Columbus, OH, USA. He is currently a Full Professor with the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, China. His research interests include speech processing, underwater acoustic signal processing, machine learning, statistical signal processing, and artificial intelligence. He is a Member of IEEE SPS and ISCA.

Xuelong Li (Fellow, IEEE) is currently the CTO and Chief Scientist of China Telecom, Beijing, China, and he founded China Telecom Institute of Artificial Intelligence (TeleAI), Beijing.