https://doi.org/10.63313/iscia.69022

Research on DOA estimation based on Transformer model

ChenXin Wang
Zhejiang University of Technology, Hangzhou, China
Email: cxwang@zjut.edu.cn

How to cite this paper: Wang, C. X. (2025). Research on DOA estimation based on Transformer model. In *Proceedings of the 2025 6th International Symposium on Computer Engineering, Information Science & Application Technology (ISCIA 2025)* (pp. 120–142)
Published: 2025-12-29

Abstract

This study primarily aims to validate the applicability of Transformer in DOA estimation and to explore data processing procedures suitable for this field. Since the experiments are based on simulation data, the study first examines the size of the dataset and its parameters. Given that training Transformer networks typically requires larger datasets and is more challenging compared to convolutional neural networks, a convolutional neural network was chosen for testing. After selecting the dataset, the study presents a DOA estimation model implemented using Transformer and analyzes its performance. Finally, it provides methods for DOA estimation under conditions of unknown source numbers and wideband conditions.

Keywords

Transformer model; DOA estimation; simulation data; convolutional neural network

1. Introduction

The direction of arrival (DOA) estimation is a key issue in the field of array signal processing. Essentially, DOA estimation involves estimating the direction of electromagnetic waves or signals based on the received signals from an array. Over the past two decades, research on DOA estimation has been a focal point across various fields, and its theoretical findings have been widely applied in sonar, radar, radiation source localization, military communications, and other areas.

1.1. The DOA estimates the history of development

In the late 1940s, Bartlett introduced the first array-based DOA estimation algorithm, known as Conventional Beamforming (CBF). This algorithm leveraged the similarity between time-domain and spatial spectrum estimates to extend the time-domain Fourier transform method to the spatial domain. However, under limited aperture conditions, the accuracy of DOA estimation was constrained by the

Rayleigh limit.[8] In 1979, Schimidt R.O. and colleagues developed the Multiple Signal Classification (MUSIC) algorithm, which is a cornerstone of subspace classification algorithms.[15] This algorithm uses the orthogonality between signal and noise subspaces to obtain the spatial spectrum. The MUSIC algorithm is computationally intensive and susceptible to noise interference. To reduce computational load, Barabell proposed the one-dimensional Root MUSIC algorithm, which uses the roots of polynomials formed by the noise subspace to estimate the DOA of signals without the need for spatial scanning, thus reducing computational complexity. However, this algorithm is only suitable for uniform linear arrays.[9] In the late 1980s, Roy and colleagues introduced the ESPRIT algorithm, which, based on subspace rotation invariance techniques, eliminates the need for full spatial domain searches, thereby improving computational efficiency.[2] However, this approach also reduces DOA estimation accuracy and is more susceptible to noise. To address the issue of subspace algorithms being susceptible to noise, Viberg and Ziskind proposed the weighted subspace fitting algorithm and the maximum likelihood algorithm by leveraging the fitting relationship between array manifold matrices and signal subspaces. These algorithms perform well in DOA estimation under conditions of low SNR and limited sampling. However, the weighted subspace fitting algorithm requires searching across multiple dimensions, while the maximum likelihood algorithm employs iterative methods, both of which result in high computational complexity. This indicates that traditional DOA estimation algorithms struggle to achieve both low computational complexity and high resolution simultaneously, necessitating a trade-off. Additionally, most traditional DOA estimation algorithms assume that the array manifold is known and that the noise is Gaussian white noise, meaning the noise covariance matrix is a multiple of the identity matrix. However, real-world applications often involve unknown factors such as array errors and Gaussian spatial correlation noise, making it challenging to meet these assumptions and leading to a decline in the performance of traditional DOA estimation algorithms.[3]

1.2. Research status of DOA estimation based on deep learning

With the rapid advancement of deep learning, an increasing number of researchers are applying it to the Direction of Arrival (DOA) estimation. Deep learning can learn key information from array signals, forming a unique mapping relationship between features and DOA, thus determining the direction of the signal's arrival. In 2015, Xiong Xiao and colleagues first applied a single-layer deep neural network to DOA estimation. The input to the deep neural network was the lower triangular elements of the generalized cross-correlation matrix of the received signal. They used a multi-layer perceptron to learn the nonlinear mapping relationship between signal features and DOA. Experiments showed that the algorithm's performance outperformed traditional methods in most conditions, marking the beginning of research into DOA estimation algorithms based on deep learning. Subsequently,

Takeda and colleagues improved upon this by using a multi-hidden-layer deep neural network for DOA estimation, incorporating the concept of hierarchical ensemble training. This approach outperformed conventional deep neural network-based DOA estimation algorithms, further advancing the field of deep learning in DOA estimation. In 2017, Chakrabarty and Habets first used a convolutional neural network to solve the DOA estimation problem. Experiments demonstrated that this algorithm had high DOA estimation accuracy and was robust against microphone position perturbations. In 2018, Liu Zhangmeng and colleagues proposed a DOA estimation algorithm based on deep neural networks. They first utilized a multi-task autoencoder to spatially divide the source signal into several smaller subregions. Then, the output of each autoencoder is fed into the corresponding parallel classifier. Through this process, the algorithm's generalization ability is significantly enhanced compared to DOA estimation algorithms based on radial basis functions and support vector machines. In the same year, Huang et al. introduced a new framework that integrates large-scale multiple-input multiple-output (MIMO) systems with DOA estimation into deep learning, resulting in an algorithm that demonstrates superior performance and robustness in both DOA and channel estimation. In 2019, Wang et al. developed a deep learning-based DOA estimation algorithm that can be applied to arrays of varying microphone counts and geometries, outperforming traditional DOA estimation methods in environments with strong noise and indoor reverberation. In the same year, Wu et al. utilized the spatial sparsity of source signals in deep convolutional neural networks to enhance DOA estimation performance under low SNR conditions. In 2020, Elbir designed multiple parallel convolutional neural networks to estimate the direction of arrival (DOA) of the source signal, reducing the training data load and complexity. In the same year, Yao and colleagues introduced a DOA estimation algorithm based on recursive neural networks, capable of estimating the DOA of unknown signal sources. However, this algorithm performs poorly in scenarios with colored noise and low SNR. Unlike previous methods, this paper trains a denoising autoencoder (DAE) to predict the expected array covariance matrix, rather than the direction of the source signal. This approach allows traditional DOA estimation algorithms, such as MUSIC, to be applied to the estimated array covariance matrix output by the DAE, thereby enhancing performance in low SNR conditions.

To sum up, compared with traditional DOA estimation algorithm, deep learning-based DOA estimation algorithm has improved in terms of estimation performance and generalization. However, under the condition of array phase error or Gaussian spatial domain correlation noise, the generalization performance of deep learning-based DOA estimation is poor.

1.3. Main content and research ideas

This paper explores the application of deep learning in DOA estimation, tailored to

the characteristics of the direction-finding field. It begins by outlining the modeling and data processing methods for DOA estimation, as well as the algorithmic steps for applying deep learning to this task. Next, it examines the properties of the datasets used in simulations under a single-source condition using a convolutional neural network with fewer parameters and simpler training. After determining the dataset, the paper introduces a Transformer model suitable for DOA estimation along with its corresponding training scheme. Finally, it presents DOA estimation methods under varying signal source numbers and wideband conditions.

2. Deep learning theory foundation

Machine learning (ML) supports various sectors of modern society, such as data mining, facial recognition, and recommendation algorithms, providing convenience. Meanwhile, deep learning (DL) is widely applied due to the limitations of traditional ML in processing raw data. Cloud computing and big data have enhanced computational power and data volume, reducing training inefficiencies and overfitting risks, which has made DL popular. DL was introduced by Hinton et al. in 2006, based on artificial neural networks, using big data to train multi-layer network models, increasing layers to enhance depth and complexity. Compared to shallow learning methods like SVM, DL processes input signals layer by layer to transform them into feature representations, making complex tasks feasible. Currently, research based on deep learning has developed methods such as deep neural networks (DNN), recurrent neural networks (RNN), residual neural networks (ResNet), and long short-term memory (LSTM). Among these, DNN includes convolutional neural networks (CNN) and deep belief networks (DBN). These methods have greatly improved the level of technology in speech recognition, visual target recognition, target detection, and drug discovery and genomics. Here are some basic concepts about DNN.[1]

2.1. Activation function

The most significant breakthrough in Deep Neural Networks (DNNs) was the introduction of nonlinear activation functions. Without these functions, a neural network, regardless of its depth, would be essentially no different from a single-layer network. The role of activation functions is to introduce non-linear elements into the neural network, enabling it to better tackle more complex problems. In the early stages of neural network research, Sigmoid and tanh functions were primarily used as activation functions. Due to their bounded outputs, they are easily suitable for use as inputs to the next layer. In recent years, especially in the study of multi-layer networks, the ReLU function and its variants have become widely adopted.

2.1.1. Figure 1 shows the geometric figure of Sigmoid function, and its mathematical formula is as follows:

$$\text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$$

(Equation 1)

As shown in Figure 1, the sigmoid function is advantageous because it can map continuous input features to outputs between 0 and 1. Specifically, when the input is a large negative number, the output is 0; conversely, when the input is a large positive number, the output is 1. In early research, the sigmoid function was widely used, but due to its high probability of experiencing gradient vanishing and other inherent issues, such as non-zero mean outputs and analytical expressions containing power operations, its use has gradually decreased in recent years.

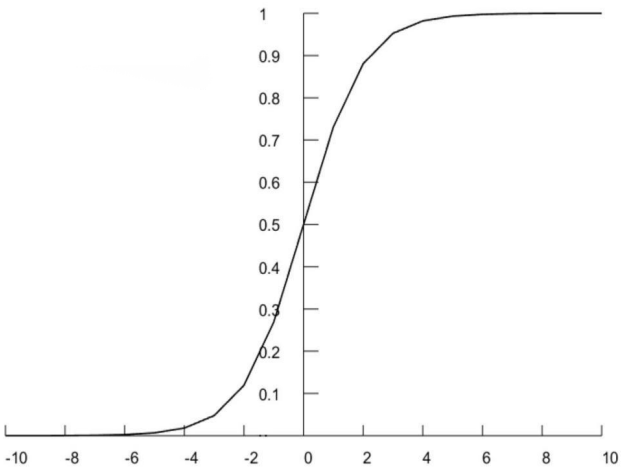


Figure 1. Sigmoid geometric graph of the function

2.1.2. Figure 2 is the geometric graph of the hyperbolic tangent function, and its mathematical expression is as follows

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

(Equation 2)

As an activation function commonly used in early neural network research, tanh function solves the problem that the output of Sigmoid function is not zero mean, but the remaining problems of gradient disappearance and power operation still exist.

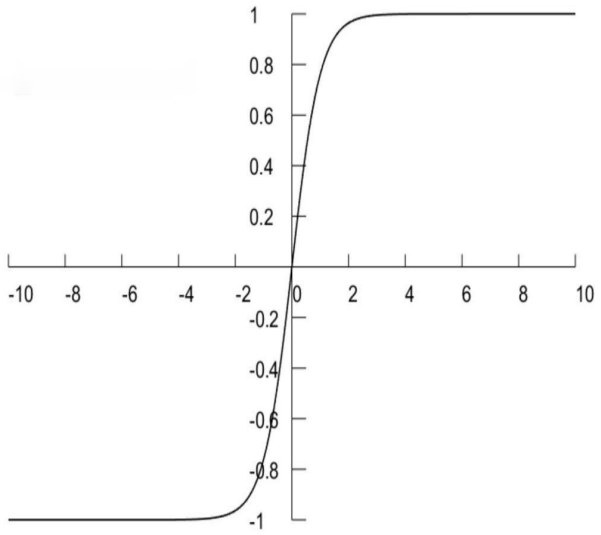


Figure 2. Geometric diagram of the hyperbolic tangent function

2.1.3. Figure 3 is the geometric figure of ReLU function, and its mathematical expression is as follows:

$$ReLU = \max(0, x)$$

(Equation 3)

From Equation 3, it is evident that the essence of the ReLU activation function is to select the maximum value. Compared to the previous two activation functions, the ReLU function excels by addressing the gradient disappearance issue within the positive domain and achieving very fast computation and convergence speeds. However, its output still does not conform to a zero mean distribution. Subsequent research has led to the development of many improved versions of the ReLU function, such as Leaky-ReLU, P-ReLU, and R-ReLU.

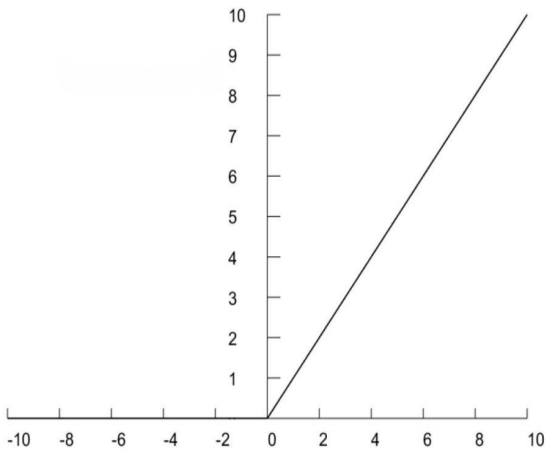


Figure 3. Geometric diagram of the ReLU function

2.2. Cost function

The loss function (Loss Function) in Deep Neural Networks (DNNs) is used to evaluate the discrepancy between the model's predictions and the actual values. In some articles, the loss function is defined as the error calculated for a single sample, while the cost function is defined as the average of errors across all samples in the entire training set. In essence, there is no strict distinction between the loss function and the cost function; both are used to measure the performance of a model. Additionally, they are key components of the objective function in neural networks. The training process of a neural network essentially involves minimizing the loss function or cost function. A smaller loss function indicates that the model's predictions are closer to the actual or expected values, thereby improving the model's accuracy. Here, we discuss several common cost functions.

2.2.1. Secondary cost function

There are many functions that can be used as cost functions, the most famous of which is the quadratic cost function. Its mathematical expression is as follows:

$$J = \frac{1}{2N}\sum_{i=1}^{N}\left\|y(x_i) - \hat{y}(x_i)\right\|^2$$

(Equation 3)

Here, J denotes the cost function, N represents the total number of samples, xi denotes the i-th sample, y (xi) denotes the true $\hat{y}$value corresponding to the i-th sample, and (xi) denotes the output or predicted value of the neural network when the i-th sample is used as input. When the number of samples is 1, Equation 3 becomes:

$$J = \frac{(y - \hat{y})^2}{2}$$

(Equation 4)

The second cost function has a relatively simple form, often appears in some regression tasks, and is suitable for linear output neurons.

2.2.2. Cross-entropy cost function

In addition to the quadratic cost function, cross-entropy function is also a relatively common cost function. Especially in the classification problems based on DNN, cross-entropy is the most commonly used one at present. Its mathematical expression is as follows:

$$J = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{K} y_{ij} \ln \hat{y}_{ij}$$

(Equation 5)

Where J represents the cost function, N represents the total number of samples, K represents the total number of categories$\hat{y}$, yij represents the true value of the j-th

category corresponding to the i-th sample, and yij represents the corresponding network output result, that is, the predicted value. Compared with the quadratic cost function, cross entropy cost function is more suitable for some multi-classification problems.

In addition to the two cost functions mentioned above, deep neural networks can also use log-likelihood and absolute value cost functions. Different practical problems require different cost functions to ensure the network converges quickly. When selecting a specific cost function, the corresponding activation function should also be considered. For instance, when using the Sigmoid function as the activation function, it is advisable to use the cross-entropy cost function instead of the quadratic cost function, which can enhance the convergence speed of the DNN algorithm. If the softmax activation function is chosen for regression tasks, the commonly used cost function is the log-likelihood cost function. When the DNN handles binary classification tasks, the log-likelihood cost function can be simplified into the form of the cross-entropy cost function.

2.3. Convolutional neural network

As illustrated in Figure 4, a convolutional neural network (CNN) comprises a convolutional layer, a pooling layer, and a fully connected layer. The CNN uses convolutional operations to extract structural information from the data. [7] The pooling layer performs downsampling on the input data, reducing computational load. The fully connected layer extracts abstract features from the outputs of the convolutional and pooling layers and establishes a mapping relationship between these features and the results. The fully connected layer contains a large number of parameters, which can be adjusted through backpropagation to minimize the network's loss function, thereby enhancing the model's performance.[4]

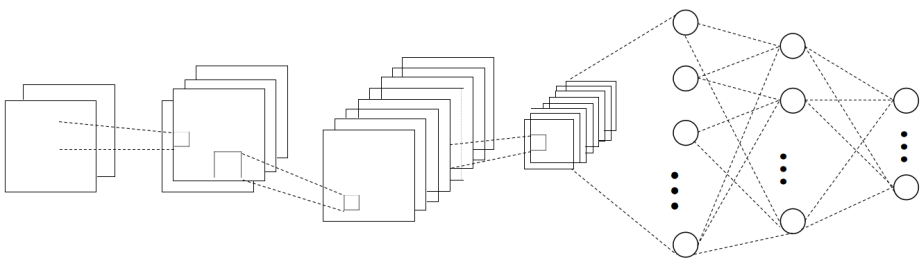


Fig 4. Convolutional neural network structure

The size of the output data from a convolutional layer is determined by the dimensions of the input data and the convolution kernel. The input to the convolutional layer is a three-dimensional tensor, denoted as W×H×C, where W and H represent the width and height of the input data, and C represents the number of channels in the input tensor. During the convolution operation, the convolution kernel slides from the top-left corner of the input data, multiplies and sums with the

corresponding data points to obtain the value of the first row and first column of the output data. It then slides rightward at the kernel's step size, sequentially obtaining the values of the output. Figure 3.3 illustrates a 5×5×1 tensor, which, after passing through a 2×2×1 convolution kernel, outputs a 4×4×1 tensor. Through convolution operations, features can be extracted from the data, enabling neural networks to better learn these features and reducing the computational load on the network.[5]

2.4. Transformer model

Since its introduction in 2017, the Transformer model has achieved significant success across a range of NLP tasks, including machine translation, text classification, sentiment analysis, and question-answering systems. Furthermore, pre-trained models based on Transformers, such as BERT and the GPT series, have also excelled in various downstream tasks, significantly advancing the field of natural language processing.

The structure of Transformer is illustrated in Figure 5, where the left and right sides represent the encoder and decoder architectures, respectively. Both use stacked self-attention and fully connected layers. The encoder consists of N identical layers, each containing two sub-layers: the first is a multi-head self-attention layer, and the second is a simple fully connected network. In each sub-layer, residual connections are applied before Layer Normalization. Specifically, the output of each sub-layer is LayerNorm (x + Sublayer (x)), where Sublayer (x) represents the function implemented by the sub-layer. To facilitate the implementation of residual connections, all sub-layer outputs have a dimension of dmodel.

The decoder is also composed of N identical layers. In addition to the two sub-layers in each encoder layer, a third sub-layer is added to the decoder for self-attention processing of its output. The encoder uses the same residual connection as the decoder.

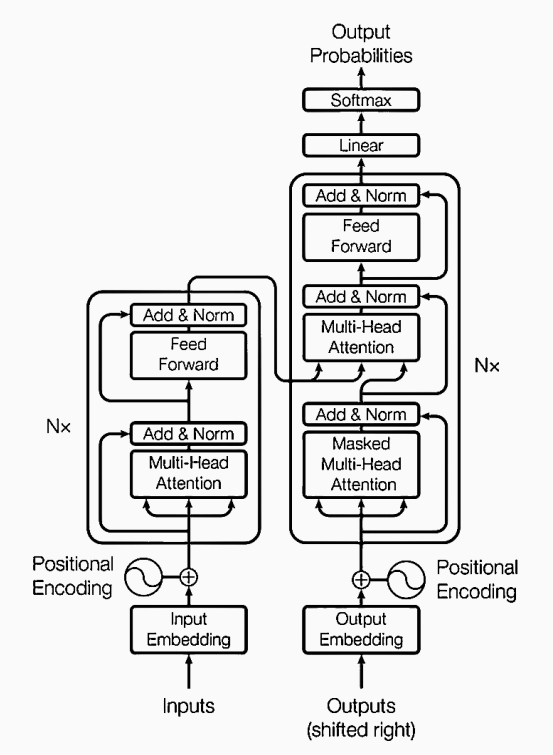


Figure 5. Transformer model structure

3. DOA estimation framework in deep learning

Figure 6 shows the workflow of training a DL-based DOA estimation model.[9]

Data: D = {($\boldsymbol{x}(1)$, $\boldsymbol{y}(1)$), … , ($\boldsymbol{x}(D)$, $\boldsymbol{y}(D)$)} Result: Neural network for DOA estimation 1 Set the current task objective, such as end-to-end, only for feature extraction, or for a specific scenario or performance, etc;
2 Select the model structure, such as CNN, Transformer, etc;
3 Set hyperparameters, including model depth, structural parameters, activation function, BatchSize, Dropout, learning rate, etc;
4 Train the network on the training set using the BP algorithm.;
5 Test the model performance through the validation set.;
6 If the model needs improvement then
7 Return to step 2 or 3;
8 else
9 Determine the goals for the next stage and return to step 1;
10 end

Figure 6. Flow of training neural networks for DOA estimation

3.1. Problem modeling and data processing

The problem of using neural networks for angle measurement can be categorized into regression and classification problems. In regression, the goal is to directly predict the value of the angle.[11] However, in DOA estimation, the number of sources is often unknown and can vary, necessitating the selection of an appropriate output format to handle this issue. For classification, pre-set angle grids are

required, and the probability of a signal appearing at each grid point is determined based on the array signals. This approach is similar to that used in MUSIC and compressed sensing algorithms. Regarding the input form of the network, the author explored two different methods: the first method uses the correlation matrix as the input; the second method, inspired by the commonly used embedding (Embedding) input format in natural language processing, directly uses the array signals as the network input. When using the correlation matrix as input, the matrix is converted into a real matrix, with the network input r ∈ RM×M×2. Following the conventions in image processing, the third dimension represents different channels, where the first channel is the real part of the matrix, and the second channel is the imaginary part, i.e., r [1] = Real{R}, r [2] = Imag{R}. Finally, r needs to be normalized

$$\tilde{\boldsymbol{r}} = \frac{\boldsymbol{r} - \min\{\boldsymbol{r}\}}{\max\{\boldsymbol{r}\} - \min\{\boldsymbol{r}\}}$$

(Equation 6)

When the array signal is used as the input, since the array signal is $\boldsymbol{x} \in$ CM×N To convert the complex matrix into a real matrix and match the input form of the model, the real and imaginary parts in $\boldsymbol{x}^{T}$ are extracted respectively, and then concatenated to obtain:

$$\tilde{\boldsymbol{x}} = (\mathrm{Real}\{\boldsymbol{x}^{\mathbf{T}}\}, \mathrm{Imag}\{\boldsymbol{x}^{\mathbf{T}}\}) \in \mathrm{R}^{\mathrm{N}\times 2\mathrm{M}}$$

(Equation 7)

For the network's output, this paper models the DOA estimation problem as a multi-label classification problem and uses one-hot encoding to convert the angle representation into a vector representation. Let θmax represent the maximum arrival wave angle, and divide the interval [-θmax, θmax] into G angle grid points, where the vector representation of the angles is y ∈ {0,1}^G. In this paper, θmax = 45, and G = 91, meaning that 91 grid points are divided at intervals of 1 within the range [-45,45°]. For any arrival wave angle, the elements in y corresponding to the adjacent angle grid points are set to 1. For example, for θ = -39.3, its nearest angle grid point is-39, and the corresponding vector representation is y = [0, 1, 0, … , 0], y ∈ {0, 1}91. When there are multiple direction sources, only the vector representations of different angles need to be added together. When there are a total of D samples, the dataset consists of D input-output pairs:

$$\mathrm{D} = \{(\mathrm{x}^{(1)}, y^{(1)}), \ldots, (\mathrm{x}^{(\mathrm{D})}, y^{(\mathrm{D})})\}$$

(Equation 8)

3.2. Evaluation indicators

In the field of DOA estimation, root mean square error (RMSE) is usually used to evaluate the performance of the algorithm. For a single sample, the formula is:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{K} (\hat{\theta}_i - \theta_i)^2}{K}}$$

(Equation 9)

Multiple samples were used

$$\mathrm{RMSE} = \sqrt{\frac{1}{DK}\sum_{j=1}^{D}\sum_{i=1}^{K}(\hat{\theta}_i^{(j)} - \theta_i^{(j)})^2}$$

(Equation 10)

Since the DOA estimation problem is modeled as a multi-label classification problem with an accuracy of 1 in this paper, there is an error of rounding when calculating RMSE. Here, the expected value of RMSE only considering the error of rounding is given. In the case of only one sample and one source, let us denote it as

$$\mathrm{E}[\mathrm{RMSE}] = \mathrm{E}[\sqrt{(\hat{\theta} - \theta)^2}] = \mathrm{E}[|u|] = \int_{-0.5}^{0.5} |u|\, du = 0.25^\circ$$

(Equation 11)

Furthermore, the study found that when the DOA estimation problem is modeled as a multi-label classification problem, the model's output resembles a spatial spectrum. Therefore, in cases where the number of sources is unknown, it is necessary to select an appropriate threshold to determine which directions contain sources. In this study, the author uses accuracy (Accuracy) to measure the model's ability to accurately identify the number of sources, defined as the proportion of samples where the predicted number of sources matches the actual number, calculated as follows

$$\mathrm{Accuracy} = \frac{\sum_{i=1}^{D}[\mathrm{Peaks}(\hat{y}^{(i)}) = K]}{D}$$

(Equation 12)

The Peaks (·) function is used to confirm the number of sources in the model prediction results.

4. Test on a data set based on CNN

Since neural networks acquire information by learning patterns in the data, it is essential to ensure that the data contains all the necessary information for accurate DOA estimation. This section aims to investigate how noise, the number of fast shots, and the dataset size affect DL-based DOA estimation methods, thereby forming a fundamental understanding of this research area.[6]

Since CNN requires a smaller data set than Transformer, is easier to train and has become the main framework of DL-based DOA estimation methods, CNN is first used to study the data set used for model training.

4.1. Model structure

The model structure is illustrated in Figure 7. The CNN takes a normalized signal correlation matrix $\tilde{\boldsymbol{r}} \in$ CM×M×2 as input and outputs a vectorized representation of angles$\hat{\boldsymbol{y}}$ ∈ {0, 1}G. The network consists of three convolutional layers, each with 64 filters, a 3×3 kernel size, and ReLU activation function. After flattening the data, a

fully connected layer with 128 neurons and ReLU activation function is used to extract features. The final output layer is a fully connected layer with G neurons and a Sigmoid activation function. Additionally, a Batch Normalization layer is added after each hidden layer, and the Pdropout rate is set to 0.5.

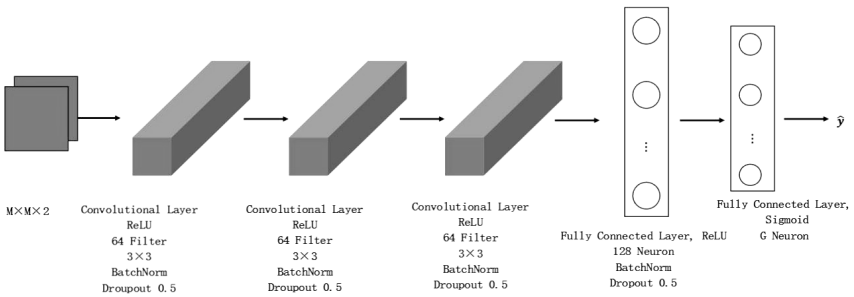


Figure 7. DOA estimation model based on CNN

4.2. Training Settings

The optimizer is set to Adam, with the following parameters: learning rate lrate = 0.001, β1 = 0.9, β2 = 0.999, and ∈ = 10^{-7}. The batch size is set to 512, epochs are set to 200, and the loss function is binary crossentropy, defined as:

$$J = -\frac{1}{DG}\sum_{j=1}^{D}\sum_{i=1}^{G}[\boldsymbol{y}_i^{(j)}\log\hat{\boldsymbol{y}}_i^{(j)} + (1-\boldsymbol{y}_i^{(j)})\log(1-\hat{\boldsymbol{y}}_i^{(j)})]$$

Equation 13

4.3. Simulation results

The author tested the impact of various properties and scales of the simulation dataset on training outcomes and model performance. In the experiment, the array model was a uniform linear array with M = 16 elements, carrier frequency f = 2GHz, element spacing d = λ/2, and the source angle was randomly selected within the range [-45.,45.] degrees. This section focuses on the scenario where the number of sources K = 1.

4.3.1. Training set noise test

First, test whether the dataset used for training the network should contain errors. Two sets of different training data are generated through simulation: one set contains array signals without noise, while the other set includes additive Gaussian white noise, with the signal-to-noise ratio (SNR) randomly selected within the range [-20,30] dB. Each set has samples of D=10000, and each uses 100 fast shots. Two different models are trained using these two sets of data. Unlike when the number of

sources is unknown, here we assume that the model knows the number of sources K = 1, meaning the estimated angle is the angle corresponding to the category with the highest probability in the model's output. Monte Carlo trials were repeated 1,000 times, and the performance of the two networks was compared at different SNR levels, as shown in Figure 8.

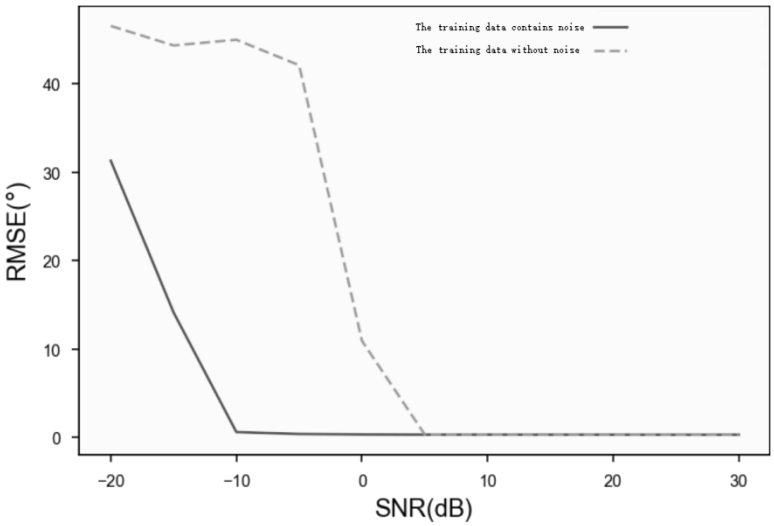
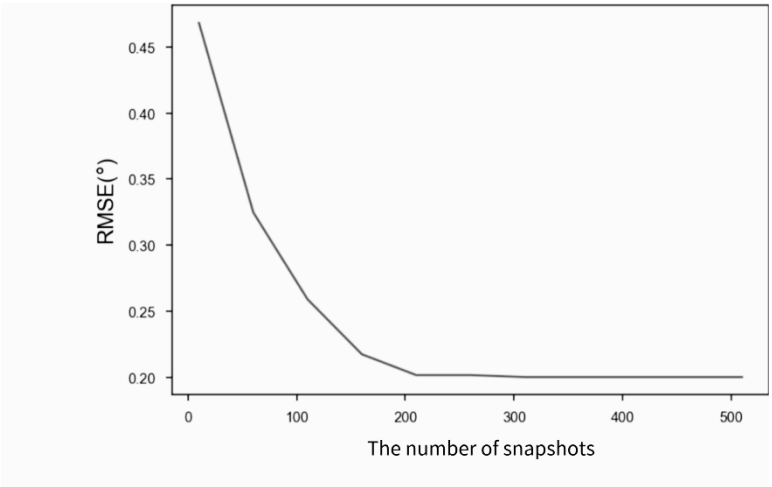


Figure 8. The influence of noise in training data on model performance

As shown in Figure 8, when the training data is free of noise, the model's performance drops sharply at 0dB. When the SNR falls below 0dB, the model's RMSE reaches approximately 45, indicating that the model has lost its predictive capability. However, when the training data contains noise, the model can still achieve high-precision predictions. At a SNR of-10dB, the RMSE is 0.7, and it stabilizes at around 0.3 once the SNR reaches 10dB. After the SNR drops below-10dB, the model's RMSE begins to rise slowly but remains lower than that of the model trained on noise-free data. This suggests that the noise in the data acts as a regularizer, preventing overfitting.

4.3.2. Fast shot number test

Next, we examine the impact of the number of fast snapshots on deep learning-based DOA estimation. With a source angle of 10.2 and a signal-to-noise ratio (SNR) of-5dB, we highlight the effect of the number of fast snapshots on model performance. The number of fast snapshots is selected in steps of 50, ranging from 10 to 510, with both training and test data using the same number of fast snapshots. A probability threshold of 0.2 is set, and 1000 Monte Carlo experiments are conducted, as shown in Figure 9.



The number of snapshots

Figure 9. Influence of fast shooting number on the performance of convolutional model

It can be seen that the ranging error decreases with the increase of the number of snapshots. When the number of snapshots is 100, the RMSE is 0.26, and when the number of snapshots is more than 200, the RMSE remains at about 0.2. The results show that under the condition of SNR = -5dB, the model only needs 200 snapshots to completely eliminate the influence of noise.

4.3.3. Data set size test

Finally, the impact of the number of test samples on model performance was investigated. Different sizes of datasets were generated through simulations and divided into training and validation sets in a ratio of 0.85:0.15. The signal-to-noise ratio (SNR) was randomly selected within the range of [0,30] dB, with a fast sampling rate of N = 200. It is assumed that the model already knows the number of sources K = 1. The results on the validation set are shown in Figure 10. The experimental results indicate that when the sample size increased to 1300, the model's root mean square error (RMSE) decreased to 0.75°. When the sample size increased to 5000, the RMSE reached its optimal value, stabilizing at 0.25°.

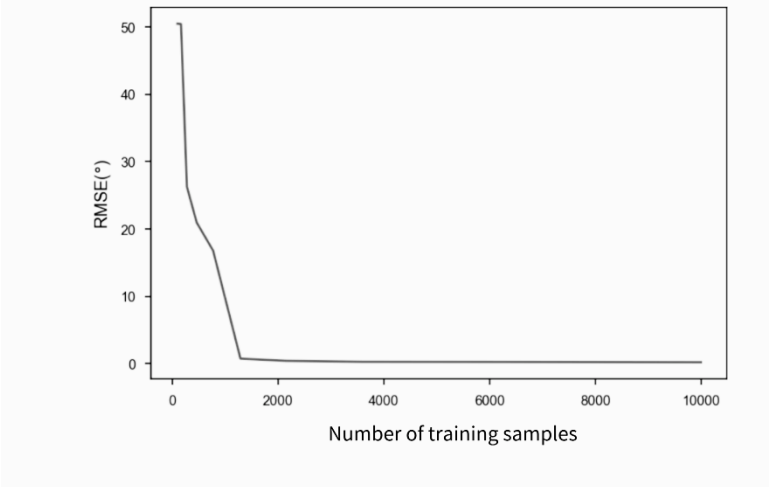


Figure 10. Influence of data set size on convolutional model performance

The simulation results show that the model trained with the dataset provided in this section can accurately predict DOA (Direct Of Arrival) estimation. This highlights the importance of the research on the training dataset, indicating that the training set should include noise, fast snapshots, and samples distributed similarly to those in the test set, with a sufficient scale. Under single-source conditions, when the number of fast snapshots is less than 200, the model performance starts to decline slowly; the impact of sample size on model performance shows a stepwise effect: when the sample size is less than 1300, performance deteriorates rapidly, but after 1300, the model performance improves gradually as the sample size increases, stabilizing at the theoretical optimal value once the sample size reaches 5000.

5. DOA estimation method based on Transformer

The Transformer model has achieved significant success in the field of Natural Language Processing (NLP), thanks to its advantages such as long-distance dependence and parallelization. Research indicates that when the dataset size and model parameters are sufficiently large, the Transformer model typically performs optimally. However, multi-source and wide-band DOA estimation based on deep learning (DL) often requires larger datasets compared to single-source and single-frequency scenarios. Therefore, this paper focuses on integrating the Transformer model into the DOA estimation domain.[13]

5.1. Data set

This paper focuses on the scenario where the number of sources K is 1. A uniform linear array with 16 elements is selected, with an element spacing d = λ/2 and a carrier frequency f = 2GHz. The source angles are randomly generated within the range of [-45,45°]. Based on the test results from Chapter 4.3, the signal-to-noise ratio (SNR) is randomly selected within the range of-20 to 30dB, and the number of

fast samples N is set to 100. A total of 10,000 samples are generated, which are randomly divided into a training set and a validation set in a ratio of 0.85:0.15.[14]

5.2. Model structure

As illustrated in Figure 12, the model takes a real-valued signal matrix as input $\tilde{\boldsymbol{x}} \in$ RN×2M and outputs a vectorized representation of angles, denoted as $\hat{\boldsymbol{y}} \in$ RG×1. The input data is first linearly projected through a fully connected layer with dmodel neurons, which does not use an activation function, resulting in a data dimension of N × dmodel. This is followed by NA identical layers, each consisting of two sub-layers: a multi-head self-attention layer and a multi-layer perceptron layer, both incorporating residual connections. In the multi-head self-attention sub-layer, there are h heads, while the multi-layer perceptron sub-layer comprises two fully connected layers, each with 2 × dmodel and dmodel neurons, respectively, and both layers use the GELU activation function. Finally, the flattened data is sequentially fed into fully connected layers with dfc and d = G neurons, where the latter uses the Sigmoid activation function.

In addition, dropout with probability Pdrop1 is used in the multi-head self-attention sublayer and multi-layer perceptron sublayer, and dropout with probability Pdrop2 is used in the Flatten layer and the next full connection layer.

structure in Figure 12, the network structure parameters used in this section of the experiment are shown in Figure 11.

d_{model}	h	N_A	d_{fc}	P_{drop1}	P_{drop2}	$\frac{Parameter}{Quantity}(10^4)$
16	4	4	128	0.2	0.5	22.6

Figure 11. Network structure parameters

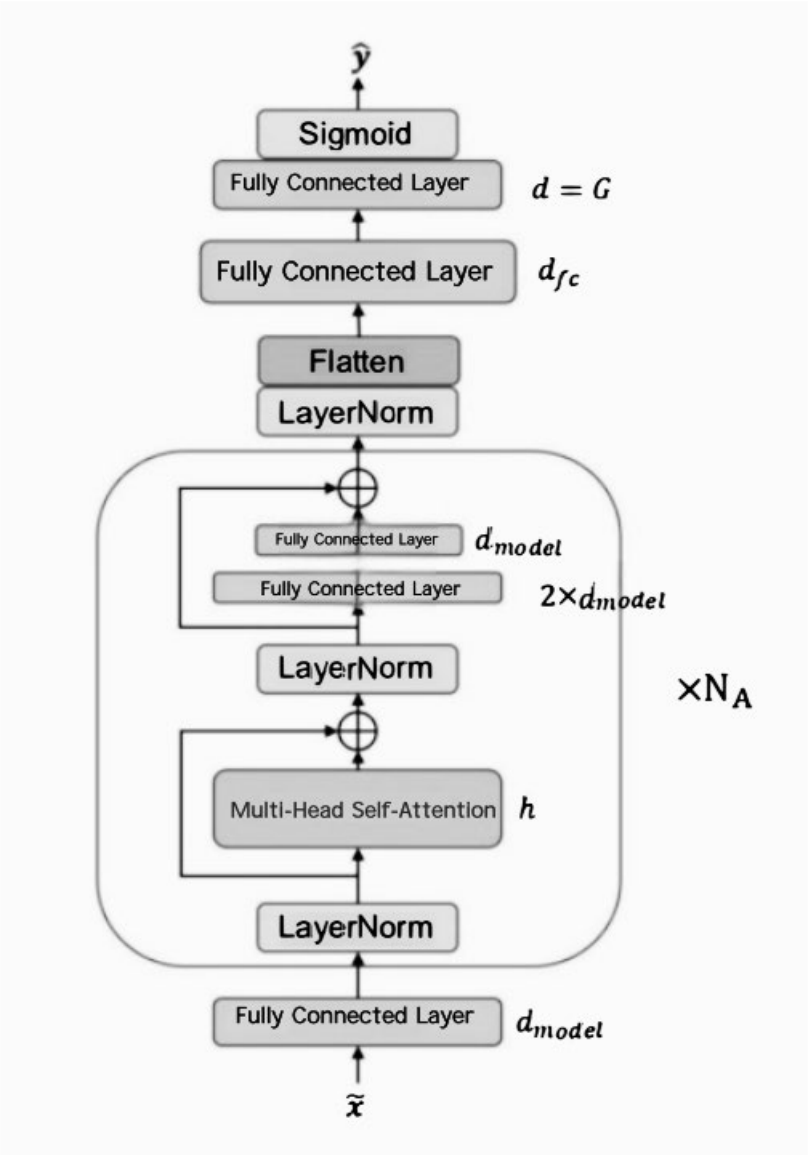


Figure 12. DOA estimation model based on Transformer

5.3. Training Settings

The optimizer used is Adam, and the parameters are set to learning rate 0.001, β 1 = 0.9, β2 = 0.999 and ∈ = 10—7. The BatchSize is selected to be 512, epoch is 100, and the loss function is binary crossentropy.

5.4. Simulation results

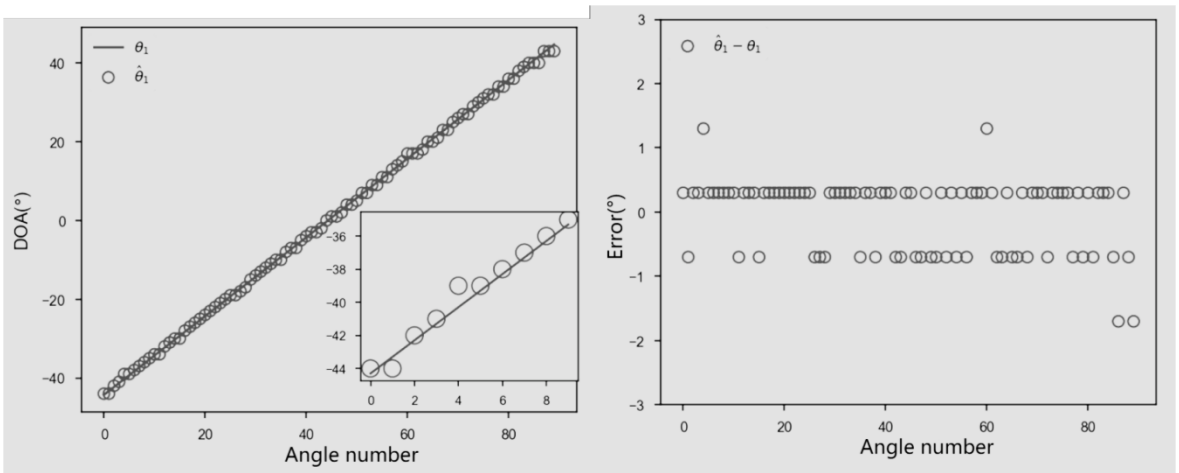
5.4.1. Model performance score

The output decision threshold was selected as 0.2, the SNR of the test data was

randomly selected within 0~30dB, the source Angle was randomly selected within [-45°,45°] and the fast shot number N = 100. The RMSE of the model was 0.51 and the accuracy rate was 98.33% after 1000 repeated experiments.

5.4.2. DOA estimation performance varies with the source angle

Two different scenarios were used for testing. First, consider a signal-to-noise ratio (SNR) of-5dB, with the signal's angle varying from-44.3 to 44.7 at intervals of 1° (angles numbered from 0 to 89). Figure 13 shows the network's predicted DOA and the corresponding errors. The RMSE for all predictions in the figure is 0.61°. It was observed that most angles were accurately predicted, while a few edge-angle errors contributed significantly to the RMSE. This is because fewer samples were used for training at the edge angles compared to the middle angles. Increasing the range of angles used for training can resolve this issue.



(a) Prediction results and real Angle; (b) Error. The initial Angle is-44.3°, SNR = -5dB

Figure 13. Comparison between the model prediction results and the real Angle when the source Angle changes continuously

5.5. DOA estimation performance varies with SNR

The signal direction is fixed at 15.2, and the SNR is sequentially-20, -15,..., 25,30dB. The fast shot number N = 100, and the Monte Carlo experiment is repeated 1000 times at each SNR level. The variation of RMSE with SNR is shown in 14

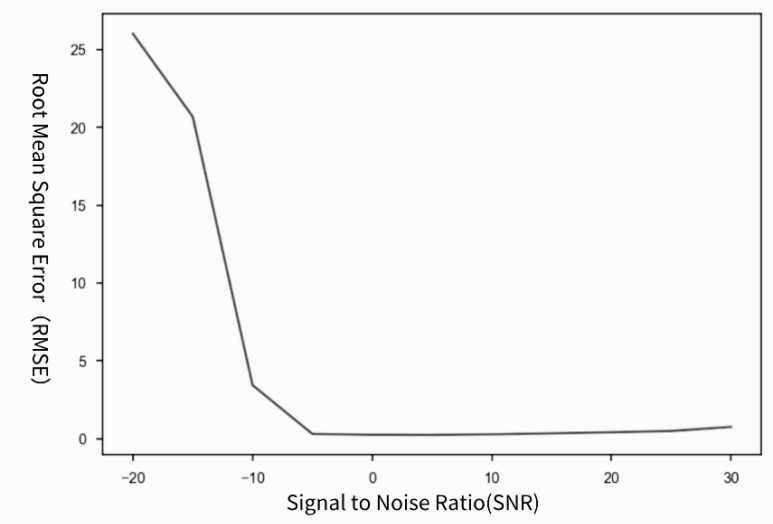


Figure 14. RMSE versus SNR

Figure 14 shows that when the signal-to-noise ratio (SNR) is no less than-5dB, the model's root mean square error (RMSE) remains around 0.3°. When the SNR is below-5dB, the RMSE increases as the SNR decreases. Additionally, a slight upward trend is observed on the far right curve of the figure. This is because the training samples with an SNR of about 30dB are only about half as many as those with an SNR of about 10dB. Increasing the range of SNRs in the data can reduce the impact of this error.

5.6. DOA estimation performance varies with the size of the training set

The impact of the size of the training set on Transformer performance. Different sizes of datasets were generated through simulation, with a sample ratio of 0.85:0.15 for the training and test sets. The signal-to-noise ratio (SNR) was randomly generated within the range of-15dB to-5dB, and the angle was randomly generated within the range of [-45,45°]. The number of snapshots N= 100. Assuming the model is known with a source K=1, the simulation results on the test set are shown in Figure 15.

The figure shows that the model performance gradually improves as the number of training samples increases. When the number of training samples is 100,000, the RMSE for the Transformer model is 2.11°, and for the CNN model, it is 2.02°. When the sample size increases to 1,000,000, the Transformer model's RMSE drops to 1.90°, while the CNN model's RMSE remains at 1.95°. When the sample size is small, the Transformer model performs worse than MUSIC. However, when the sample size reaches 1,000,000, the Transformer model's performance begins to surpass that of the CNN model, confirming that the Transformer model requires more data.

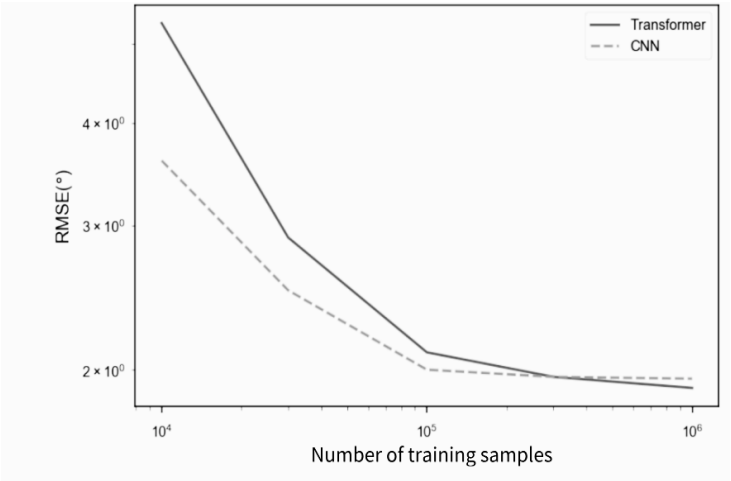


Figure 15. The effect of data set size on Transformer model performance

6. Sum up

This paper explores the DOA estimation method based on the Transformer model, examines its applicability, and proposes solutions for scenarios with unknown source numbers and wideband conditions. The estimation of DOA is a core issue in array signal processing, aimed at determining the direction of the signal source from received signals. Traditional DOA estimation algorithms struggle to balance high resolution and low computational complexity, and are often affected by array errors, noise, and other unknown factors in practical applications, leading to performance degradation. In recent years, deep learning technology has provided new approaches to DOA estimation, but existing methods perform poorly under conditions of array phase errors or non-ideal noise. Therefore, researching how to enhance DOA estimation performance using the Transformer model holds significant theoretical and practical value. First, the applicability of the Transformer model in DOA estimation needs to be verified, especially when the dataset size is small, as training it is more challenging than CNNs. Second, the generation and preprocessing of simulation data significantly impact model performance, requiring careful design to ensure data quality. Additionally, achieving high-precision DOA estimation under conditions of unknown source numbers and wideband conditions further complicates algorithm design. The Transformer consists of an encoder and a decoder, each layer including multi-head self-attention and fully connected layers, and employs residual connections and Layer Normalization. The encoder and decoder each consist of N identical layers, ensuring that the prediction at position i depends only on the output less than i. The mask operation enhances the model's ability to process sequences.

Experiments based on the Transformer model have demonstrated that the proposed network model exhibits excellent performance. The Transformer model, with its parallelization and long-distance dependency capabilities, has the potential to be

applied to large-scale datasets. Research indicates that deep learning methods, particularly the Transformer model, excel in DOA estimation, effectively addressing challenges in scenarios with unknown source numbers and wideband conditions. By designing the model structure and training strategies appropriately, the accuracy and robustness of DOA estimation can be significantly enhanced. Compared to traditional methods, deep learning models can automatically learn signal features, forming more complex mapping relationships, thereby improving estimation accuracy. This paper adds data corresponding to different source numbers to the dataset, enabling the network to predict the number of sources and perform DOA estimation. Simulation results show that the model performs exceptionally well on test data. Finally, experiments have shown that adding frequency information to the model input can solve the wideband DOA measurement problem. Future research could further explore the application of Transformer in multi-source and wideband scenarios, optimizing the model structure to meet different task requirements. Additionally, verifying the model's performance in array errors and spatially correlated noise environments, considering actual engineering conditions, will be a key research direction.

Reference

[1] Ge S, Li K, Rum S N B M. Deep learning approach in DOA estimation: a systematic literature review[J]. Mobile Information Systems, 2021, 2021: 1-14.

[2] Rumelhart D E, McClelland J L. Learning internal representations by error propagation[M]. MIT Press, 1987, 318-362.

[3] Ackley D H, Hinton G E, Sejnowski T J. A learning algorithm for boltzmann machines[J]. Cog- nitive Science, 1985, 9(1): 147-169.

[4] LeCun Y, Bengio Y, et al. Convolutional networks for images, speech, and time series[J]. The Handbook of Brain Theory and Neural Networks, 1995, 3361(10): 1995.

[5] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7): 1527-1554.

[6] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017, 60(6): 84-90.

[7] onneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image seg- mentation[C]. Medical Image Computing and Computer-Assisted Intervention, Cham, 2015: 234-241.

[8] Schmidt R. Multiple emitter location and signal parameter estimation[J]. IEEE Transactions on Antennas and Propagation, 1986, 34(3): 276-280.

[9] Rao B D, Hari K S. Performance analysis of root-MUSIC[J]. IEEE Transactions on Acoustics, Speech, and Signal Processing, 1989, 37(12): 1939-1949.

[10] Kase Y, Nishimura T, Ohgane T, et al. DOA estimation of two targets with deep learning[C]. Workshop on Positioning, Navigation and Communications (WPNC), 2018: 1-5.

[11] Liu Z M, Zhang C, Philip S Y. Direction-of-arrival estimation based on deep neural networks with robustness to array imperfections[J]. IEEE Transactions on Antennas and Propagation, 2018, 66(12): 7315-7327.

[12] Zheng W Q, Zou Y X, Ritz C. Spectral mask estimation using deep neural networks for inter-sensor data ratio model based robust DOA estimation[C]. IEEE International Conference on Acoustics, Speech and Signal Processing, South Brisbane,

QLD, Australia, 2015: 325-329.[41]S Pazos, M Hurtado, C Muravchik. DOA estimation using random linear arrays via compressive sensing[J]. IEEE Latin America Transactions, 2014, 12(5): 859-863.

[13] L Gan, J F Gu, P Wei. Estimation of 2-D DOA for noncircular sources using simultaneous SVD technique[J]. IEEE Antennas and Wireless Propagation Letters, 2008, 7: 385-388.

[14] Chakrabarty, Habets. Multi-Speaker DOA estimation using deep convolutional networks trained with noise signals[J]. IEEE Journal of Selected Topics in Signal Processing, 2019, 13(1): 8-21.

[15] S V Schell, R A Calabretta, W A Gardner, et al. Cyclic MUSIC algorithms for signal-selective direction estimation[C]. International Conference on Acoustics, Speech, and Signal Processing, Glasgow, UK, 1989, 4: 2278-2281.