Wireless Communications and Mobile Computing

Machine Learning for Ubiquitous Internet of Things 2022

Lead Guest Editor: Xiaojie Wang
Guest Editors: Amr Tolba and Zhaolong Ning

Machine Learning for Ubiquitous Internet of Things 2022

Wireless Communications and Mobile Computing

Machine Learning for Ubiquitous Internet of Things 2022

Lead Guest Editor: Xiaojie Wang
Guest Editors: Amr Tolba and Zhaolong Ning

Contents

Pillar-Based 3D Object Detection from Point Cloud with Multiattention Mechanism
Xin Li, Bifa Liang, Jinhao Huang, Yuyang Peng, Yier Yan, Jun Li, Wenli Shang, and Wei Wei
Research Article (10 pages), Article ID 5603123, Volume 2023 (2023)

A Dueling Deep Recurrent Q-Network Framework for Dynamic Multichannel Access in Heterogeneous Wireless Networks
Haitao Chen, Haitao Zhao, Li Zhou, Jiao Zhang, Yan Liu, Xiaoqian Pan, Xingguang Liu, and Jibo Wei
Research Article (14 pages), Article ID 9446418, Volume 2022 (2022)

A 3D REM-Guided UAV Path Planning Method under Communication Connectivity Constraints
Xingguang Liu, Li Zhou, Xiaoying Zhang, Xiang Tan, and Jibo Wei
Research Article (11 pages), Article ID 7410708, Volume 2022 (2022)

MANet: End-to-End Learning for Point Cloud Based on Robust Pointpillar and Multiattention
Xingli Gan, Hao Shi, Shan Yang, Yao Xiao, and Lu Sun
Research Article (12 pages), Article ID 6909314, Volume 2022 (2022)

Fault Diagnosis to Nuclear Power Plant System Based on Time-Series Convolution Neural Network
XianLing Li, DongJiang Han, XinFa Dai, ShuYu Lv, Mo Tao, Wei Zheng, and YiBin Tang
Research Article (14 pages), Article ID 3323239, Volume 2022 (2022)

Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 5603123, 10 pages
https://doi.org/10.1155/2023/5603123

Research Article

Pillar-Based 3D Object Detection from Point Cloud with Multiattention Mechanism

Xin Li,[1] Bifa Liang,[1] Jinhao Huang,[1] Yuyang Peng,[2] Yier Yan,[1] Jun Li,[1] Wenli Shang,[1] and Wei Wei[1]

[1]*Research Center of Intelligent Communication Engineering, School of Electronics and Communication Engineering, Guangzhou University, Guangzhou 510006, China*
[2]*School of Computer Science and Engineering, Macau University of Science and Technology, Taipa, 999078 Macau SAR, China*

Correspondence should be addressed to Jun Li; lijun52018@gzhu.edu.cn

Received 6 September 2022; Revised 20 October 2022; Accepted 27 January 2023; Published 9 February 2023

Academic Editor: Xiaojie Wang

Object detection in point clouds is a critical component in most autonomous driving systems. In this paper, in order to improve the effectiveness of image feature extraction and the accuracy of detection of point clouds, a pillar-based 3D point cloud object detection algorithm with multiattention mechanism is proposed, which includes three attention mechanisms SOCA, SOPA, and SAPI. The results show that the recognition accuracy of the optimized algorithm for cars, pedestrians, and cyclists on KITTI dataset is significantly improved on the detection benchmarks of BEV and 3D. Despite using only LiDAR, our algorithm outperforms PointPillars, which is one of the state-of-the-art algorithms for 3D object detection, with respect to both 3D and BEV view KITTI benchmarks while maintaining a relatively competitive speed.

1. Introduction

Machine learning (ML) and the Internet of Things (IoT) can be applied in almost every industry [1, 2], from implementing AI digital assistants to supply chain automation. With the development of machine learning technologies, autonomous driving [3] has become more and more popular. For autonomous driving, the significance of IoT and autonomous driving technology cannot be overstated. Information is the bridge between IoT sensors and self-driving cars. Relying on IoT sensors for information collection and analysis, self-driving cars are one step closer to large-scale applications. In self-driving cars, there are many sensors such as LiDAR, radar, cameras, and IoT devices that communicate with each other. Using various deep learning models based on convolutional neural networks (CNN), based on the data received, enables the car to learn automatically and continuously improve detection performance over time and experience.

At present, 3D point cloud data got from LiDAR is mainly used by autonomous vehicles for object detection. Compared with two-dimensional images, LiDAR can provide more reliable depth and shape information and locate objects with higher accuracy. However, due to nonuniform sampling, occlusion, and reflection in 3D space, the LiDAR point clouds are sparse and have highly variable density. The accuracy of traditional 3D object detection algorithms based on manual features often suffers as a result. In recent years, 3D point cloud object detection algorithms [4–6] based on deep neural networks have been improved to some extent in terms of accuracy as deep neural networks have shown excellent capability of feature extraction and can handle high-dimensional data. Nevertheless, there is still enough potential for improvement in the accuracy of detection results for some categories, due to the highly sparse and inherently irregular nature of point clouds. Some of earlier works employ 3D convolution approaches or adopt the methods that project point clouds into a perspective view. Perspective view is another extensively used representation of LiDAR. Following this line of research, the VeloFCN [7] and LaserNet [8] are some of the representative works. The new research favors BEV of the LiDAR point cloud, the

advantage of which is almost no occlusion. In 2018, Simony et al. [6] introduced Complex-yolo, a model that projects point clouds onto a 2D plane and employs a 2D image approach for object detection, thus speeding up inference of the network. However, the projection is limited by the sparsity of the point cloud, which prevents convolution from extracting features better. To cope with this issue, a common method transforms the point cloud raster into 3D voxel grids and encodes each voxel one by one using hand-crafted features. However, manual design not only cannot make full and effective use of the object's 3D information but also is not conducive to the application of other radars. Based on PointNet [9], an end-to-end deep neural network was proposed by Qi et al., in which point features are learned directly from point clouds. In 2018, Zhou and Tuzel [4] first proposed an end-to-end trainable network VoxelNet, a universal 3D detection framework. Different from most previous work, VoxelNet starts to learn information-rich feature representation and can simultaneously learn different feature representation from point clouds. However, the disadvantage of 3D convolution is that it is too time-consuming and is confronted with high computation complexity, leading to a slow inference speed of the network. Later on, Yan et al. [10] proposed SECOND, which reduces memory consumption and speeds up computation through sparse convolution operation. In order to use the standard 2D convolution detection pipeline to improve the inference speed, Lang et al. [11] proposed PointPillars, which encode point clouds into vertical pillars, a special division of voxels. To further improve the performance of point clouds object detection in challenging situations, TANet is presented by Liu et al. [12] in 2019, which utilizes a combination of attention mechanisms.

The so-called "attention mechanism" is a way of perception that mimics the human brain and human vision, a mechanism for focusing local information [13, 14]. The attention mechanism can dynamically select areas of attention as the task changes, which is achieved by adaptively assigning weights based on the degree of significance of the inputs. The point-wise attention mechanism proposed by TANet assigns weights on the basis of the importance of the points, but it has not considered the correlations between points, resulting in the loss of a fraction of valuable geometric information. Considering that each point within a pillar is semantically linked, we propose a new method called second order of point attention (SOPA), which links points with each other within a pillar. The experimental results show that the detection accuracy of pillar-based 3D object detection method with SOPA is better than that with point-wise attention. Particularly, for the two categories of pedestrians and cyclists, which are currently detected with relatively low accuracy and are more challenging to improve, there is an improvement for accuracy of each category by using SOPA. Similarly, in consideration of the channel-to-channel relationship, we propose second order of channel attention mechanism (SOCA) based on pillars between the backbone network and the feature extraction network stage, which can extract more effective information. In addition, taking into account that not all features in the pseudoimage have the same contribution to the detection task, we propose the spatial attention of pseudoimage mechanism (SAPI), which assigns different weights to each point in the pseudoimage with regard to the importance of the region in the pseudoimage to the task in the pseudoimage generation stage, which could lead to more accurate detection results. Compared with existing 3D point cloud object detection algorithms, a proposal that integrates these three second order attention mechanisms can achieve higher performance detection with relatively competitive speed.

2. Related Work

2.1. PointPillars Network. PointPillars [11] comprises three main phases: (1) a feature encoder network that transforms point clouds to sparse pseudo-images, (2) a 2D convolutional backbone network that converts pseudo-images into high-level representations, and (3) a detection head that detects and regresses 3D boxes.

2.1.1. Pointcloud to Pseudoimage. The space is partitioned into pillars [11], and at the same time, raw point clouds are assigned to the pillars and then converted into a form of sparse pseudoimage. Given l_0 to represent a point in the raw point cloud, which has coordinates x, y, z, and reflection intensity r. First, the input point cloud is partitioned into multiple pillar cells. And each pillar is a 3D grid obtained by dividing the point cloud in the X and Y plane in certain steps [0.16, 0.16]; then, a set of pillars A can be obtained. Each point in a pillar is encoded as a nine-dimensional vector E, which could be parameterized as $(x, y, z, r, x_0, y_0, z_0, x_d, y_d)$. Here, (x_0, y_0, z_0) represents the geometric centers of all points in pillars where the point cloud is located, and (x_d, y_d) indicates the offset of each point from the geometric center.

Random sampling is performed if there are more than $P = 32$ points in each pillar, and zero filling is employed if less than P, to ensure that the number of points in each pillar remains at 32. In this way, a feature tensor of (E, A, P) is obtained, and then the feature extraction is performed on the tensor. The original dimension E of the point cloud is 9, while the dimension F of point cloud is expanded to 64. Then, a feature tensor of (F, A, P) is acquired. The 2D feature map (F, A) is gained by performing the max pooling operation according to the third dimension. The last step is to generate a pseudoimage by a scatter calculus. Specifically, the original pillar is replaced by the (F, A) feature tensor generated in the previous step based on the pillar index value of each pillar to create a pseudoimage ($F = 64$, $H = 440$, $W = 500$). Here, W and H denote the width as well as the height of the canvas. In the process of constructing stacked pillars from the point cloud, the coordinates corresponding to each pillar are recorded. When the pseudoimage is constructed by the learned features, the pillars are filled with the corresponding learned features according to the pillar index.

2.1.2. Backbone. The backbone network is similar to that of VoxelNet [4], covering two subnetworks: the first one is a top-down structure with successively decreasing resolution, where the low resolution is responsible for extracting the

high-dimensional features, and the second network carries out the upsampling operations, in order to stitch together the features of the corresponding size. The first network is composed of three blocks, which consist of 3×3 conv layers behind followed by a Batch-Norm [15] layer and ReLU [16] layer. Here, the stride size is two, and the resolution decreases by half in the (H, W) direction after each convolutional layer. As it passed through three blocks, the resolution dropped three times, down to one-eighth. At the same time, the channel dimension expands from F to $4F$.

For the second network, after upsampling operation, the three blocks yield the same size features. Then, the features with the same resolution obtained after deconvolution are concatenated together to acquire an integrated feature.

2.1.3. Detection Head. SSD detection head [17] is employed for the 3D object detection in the final stage. Two anchors of 0 and 90 angles are put in the center of each pillar. For the calculation of IOU [18], rotating IOU [19] is the best one in terms of accuracy.

2.2. Attention Mechanism. Attention mechanism was first proposed in the field of image in the 1990s. The development of attention mechanism goes through four main phases. First, it utilizes RNN [20] and reinforcement learning to implement spatial attention. After that, Jaderberg et al. [21] proposed the STN, learning affine transformations to select important regions. In the third phase, CBAM [22] and Eca-net [23] are representative, of which the novel attention mechanism can adaptively predict underlying kernel features. In the fourth phase, self-attention is highly motivated [24]. Wang et al. were the first to introduce self-attention into computer vision and achieved great success in video understanding and object detection [25]. In recent years, as it has remarkable performance, an increasing number of studies based on attention mechanisms have emerged in the field of computer vision [26–28].

The existent attention methods can be divided into channel attention [23, 29], spatial attention [30, 31], temporal attention, and branch channel [32].

2.2.1. Point-Wise Attention. In the pillar-based 3D point cloud object detection algorithm, for the Kth pillar in the space, the global features of the points in the pillar are retained after the max-pooling process, and then the vector $I \in R^{N*1}$ can be obtained. Here, N represents the number of points. To limit the complexity of the network, only two fully connected layers are employed. The point-wise attention mechanism can be expressed by the following formula:

$$\mathrm{PA} = W_2 \delta(W_1 I). \tag{1}$$

$\delta(\bullet)$ is the ReLU activation function between two fully connected layers, where $W_1 \in R^{t \times N}, W_2 \in R^{N \times t}$ denote weights of the two fully connected layers. Here, t is the output length of the first fully connected layer as well as the output length of the second fully connected layer. $\mathrm{PA} \in R^{N*1}$ represents the point-wise attention of the points in the Kth pillar.

2.3. Contributions

(1) We propose three effective methods, including the second order of point attention mechanism (SOPA) based on pillars, the second order of channel attention mechanism (SOCA), and the spatial attention of pseudoimage mechanism (SAPI) after the stage of generating pseudoimage, to implement high-precision real-time object detection, respectively

(2) We conducted experiments on the KITTI dataset [33] and presented the latest detection results of cars, pedestrians, and cyclists on BEV, 3D, and AOS benchmarks. Our model runs at 34 Hz, while the detection accuracy of the category of cyclists and pedestrians, which is slightly low, is substantially improved by about 6% mAP (mAP on both BEV and AOS) over the other methods

(3) We performed several ablation experiments to examine the key influencing factors for achieving performance improvement

3. Multiattention Mechanism Network

The architecture of the multiattention mechanism network is demonstrated in Figure 1. The network accepts the raw point cloud as input and predicts 3D bounding boxes to identify cars, pedestrians, and cyclists. It is composed of the following stage: (1) first, the point cloud is voxelized, and then SOPA operation is performed on the point cloud, followed by a pillar feature network to convert it into the form of sparse pseudo-image. (2) Then, the generated sparse pseudoimage is subjected to SOCA operation. (3) After the backbone network, the SAPI operation is performed on the features of the output sparse pseudoimage, and finally, the 3D bounding box of the object is predicted by the detection head.

3.1. Second Order of Point Attention. As presented in Figure 2, when the point features are fed into the second order of attention module, the SOPA weight would be obtained as the output, and at this moment, the module is called SOPA module.

In the Kth pillar, all points $X^K \in R^{N \times C}$, where N represents the max number of points and C refers to the number of channels. Through the operation of a max pooling layer, a vector of the maximum values along the dimension C is obtained as $E^K \in R^{N \times 1}$, where $N \times 1$ represents a vector with N rows and 1 column. To maintain a large model capacity and further ensure the migration of representation capabilities, it is fed into a fully connected layer. Then, new vector is obtained as $Q^K \in R^{t \times 1}$, where t is the number of points after reduction through the fully connected layer W_1. Then, an activation function ReLU is to prevent the network from gradient disappearance. And then, the covariance matrix between two points in the same pillar is computed to get their correlation. This covariance matrix is expressed as $I^k \in R^{t \times t}$, where t represents the number of points in the SOPA while t refers to the number of channels in the

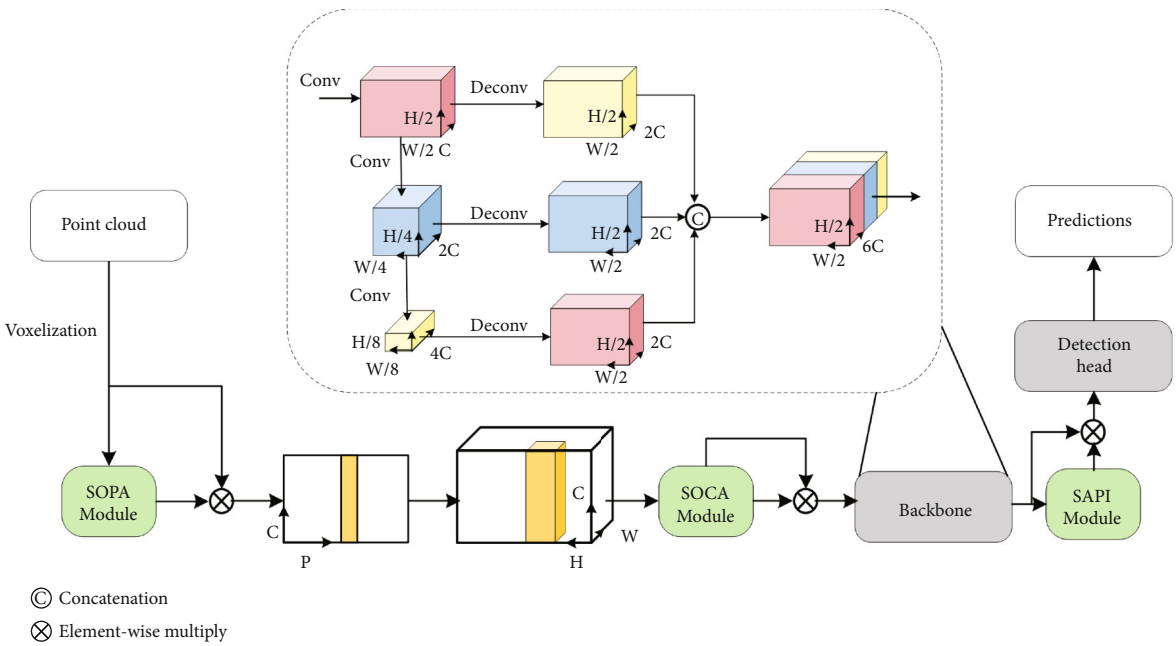


Figure 1: Network overview. The network mainly consists of SOPA, a pillar feature network, SOCA, backbone network, SAPI, and SSD detection head.

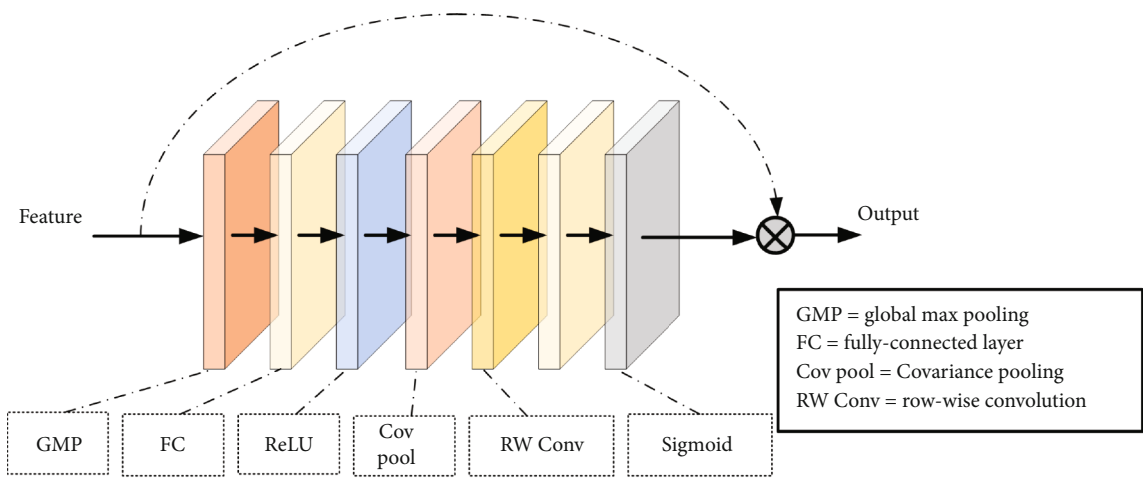


Figure 2: Second order of attention module.

SOCA, and $t \times t$ means dimension. Next, row-by-row convolution operation is applied to the covariance matrix; then, the vector is obtained as $J \in R^{t\times 1}$. The vector $J \in R^{t\times 1}$ is then fed into the fully connected layer W_2, and the N-dimensional attention vector is then gotten using a sigmoid function, denoted as $T \in R^{N\times 1}$. The SOPA can be presented as the expression as follows:

$$T = \sigma(W_2 \mathrm{RC}(\mathrm{Cov}(\delta(W_1(\mathrm{GMP}(X)))))). \quad (2)$$

Here, $\mathrm{Cov}(\bullet)$ calculates the covariance matrix of points, and $\mathrm{RC}(\bullet)$ denotes row convolution. $\delta(\bullet)$ is the ReLU activation function while $\sigma(\bullet)$ is the sigmoid function. With $W_1 \in R^{t\times N}$, $W_2 \in R^{N\times t}$ represents two different fully connected layers, respectively, and $X^K \in R^{N\times C}$ is the point in the given K^{th} pillar.

3.2. Second Order of Channel Attention. SOCA is similar to SOPA, as presented in Figure 2, when channel features are fed as inputs to the second order of attention module, the output is obtained as the weights of SOCA. For the pseudoimage features $Y \in R^{C\times H\times W}$ generated through a pillar feature network, SOCA can be expressed by the following equation:

$$M = \sigma(W_2 \mathrm{RC}(\mathrm{Cov}(\delta(W_1(\mathrm{GMP}(Y)))))), \quad (3)$$

where the superscripts H and W are the height and width of the pseudoimage.

3.3. Spatial Attention of Pseudoimage. Not all features of regions in the space can contribute equally to the task, and only the regions which are relevant to the task are of interest. Pixel points in each layer of the spatial feature are assigned with different weights. In this way, task-relevant parts of the space are chosen and then processed. Here, the spatial attention operation is performed on the pseudoimage, and that is why we refer to it as spatial attention of pseudoimage. If the feature map P and the signal H are given as input, the final output yields the spatial attention weights S. SAPI can be formulated as

$$S = \sigma\left(\varphi\left(\delta\left(\varphi_p(P)\right) + \varphi_h(H)\right)\right). \tag{4}$$

The correlation between P and H can be expressed as

$$H = \varphi_0(P). \tag{5}$$

Here, φ, φ_0, φ_p, and φ_h are computed as linear transformations with 1×1 convolution.

4. Implementation Details

4.1. Loss Function. We use the same loss functions as presented in SECOND and PointPillars. We parameterize a 3D ground truth box as $(x, y, z, w, l, h, \theta)$, where (x, y, z) represent the center location, and (w, h, l) and θ are the size and the heading angle of the bounding box, respectively. The regression residuals between ground truth and anchor boxes are computed as follows:

$$\Delta x = \frac{x^{gt} - x^a}{d^a}, \Delta y = \frac{y^{gt-y^a}}{d^a}, \Delta z = \frac{z^{gt} - z^a}{h^a}, \tag{6}$$

$$\Delta w = \log \frac{w^{gt}}{w^a}, \Delta l = \frac{l^{gt}}{l^a}, \Delta h = \log \frac{h^{gt}}{h^a}, \tag{7}$$

$$\Delta\theta = \sin\left(\theta^{gt} - \theta^a\right), \tag{8}$$

where x^{gt} denoting ground truth, and x^a is the bounding box, with $d^a = \sqrt{(l^a)^2 + (w^a)^2}$. All of the losses are summed up as the total loss of the overall network model, with the overall loss function defined as

$$L = \frac{1}{N_{\text{pos}}}\left(\beta_{\text{loc}} L_{\text{loc}} + \beta_{\text{cls}} L_{\text{cls}} + \beta_{\text{dir}} L_{\text{dir}}\right), \tag{9}$$

where N_{pos} represents the number of positive anchors. We set $\beta_{\text{loc}} = 2$, $\beta_{\text{cls}} = 1$, and $\beta_{\text{dir}} = 0.2$.

The regression loss is denoted by the following equation:

$$L_{\text{loc}} = \sum_{b \in (x,y,z,w,l,h,\theta)} \text{Smooth}L1(\Delta b). \tag{10}$$

For the classification loss, we adopt focal loss [34]:

$$L_{\text{cls}} = -a(1-p)^r \log p, \tag{11}$$

where p denotes the probability of being a positive anchor. We adopt the settings of $r = 2$ and $a = 0.25$.

5. Experiment

5.1. Dataset. All test results are evaluated using KITTI's official evaluation test metrics, including aerial view (BEV), 3D, 2D, and average orientation similarity (AOS), where AOS evaluates the average orientation similarity of two-dimensional detection (BEV). KITTI dataset are available in easy, moderate, and hard difficulties, and the official KITTI leaderboard ranked by performance on moderate. Performance is measured as mean average precision (mAP) on KITTI validation.

The experiments all employ the KITTI 3D object detection benchmark dataset, which is composed of 7,481 training samples and 7,518 test samples. And the KITTI benchmark requires detection category [33], which include cars, pedestrians, and cyclists. We also follow the generally used training-validation split, which contains 3,712 training samples and 3,769 validation samples.

5.2. Settings. Here, we use xy resolution: 0.16 m, maximum number of points per pillar (V): 100, maximum number of pillars (Z): 12000. Our approach is based on the PyTorch framework, with all networks trained on the NVIDIA 2080Ti computing platform.

We train it for 160 epochs with an initial learning rate of $2 * 10^{-4}$ and decrease the learning rate by 0.8 every 15 epochs with Adam [35] optimizer.

5.3. Results. In this section, we will introduce the results of our object detection algorithm using three types of attention mechanisms. The tables below present the effect of adding three kinds of attention mechanisms to the network. Besides, combining any of the two attention mechanisms (SOPA, SOCA) separately results in 2–3% mAP boost overall.

As shown in results Tables 1–3, the network of object detection using our second order of multiattention mechanism exceeds most of the published networks (mAP on both AOS and BEV benchmarks). As listed in results Tables 4 and 5, we also find our method combining three attention mechanisms achieved BEV mAP (88.37%, 54.13%, and 67.38%) and 3D mAP (76.22%, 49.45%, and 63.58%) in the moderate difficulty categories of car, pedestrian, and cyclist, respectively. Moreover, in most of methods using only LiDAR, better results are achieved in all categories in three difficulty cases.

We show several qualitative results in Figure 3. And while we trained only on LiDAR point clouds, the 3D bounding boxes have been projected into the camera coordinate system for the sake of clarity of interpretation. Overall, our model provides highly accurate 3D bounding boxes in all categories.

6. Ablation Experiments

In this section, we provide the results of ablation experiments to evaluate the key factors that affect the accuracy of the experiments.

Table 1: Results on the KITTI test BEV detection benchmark.

Model	Car				Pedestrian				Cyclist			
	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)
PointPillars	88.35	86.10	79.83	84.76	58.66	50.23	47.19	52.02	79.14	62.25	56.00	65.79
PointPillars + point-wise attention	89.63	86.72	83.72	86.69	58.62	53.29	48.69	53.53	83.69	63.04	61.62	69.45
PointPillars + SOPA	91.23	87.27	84.90	87.80	58.11	52.24	49.12	53.16	81.72	66.96	62.60	70.43

Table 2: Results on the KITTI test 3D detection benchmark.

Model	Car				Pedestrian				Cyclist			
	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)
PointPillars	79.05	74.99	68.30	74.11	52.08	43.53	41.49	45.70	75.78	59.07	52.92	62.59
PointPillars + point-wise attention	86.82	75.80	72.94	78.52	53.01	46.74	42.96	47.57	78.71	59.84	56.56	65.03
PointPillars + SOPA	86.13	75.27	72.86	78.08	53.44	47.61	43.45	48.10	79.34	61.63	59.18	66.72

Table 3: Results on the KITTI test average orientation similarity (AOS) detection benchmark.

Model	Car				Pedestrian				Cyclist			
	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)
PointPillars	90.19	88.76	86.38	88.44	32.39	31.41	29.84	31.21	82.43	68.16	61.96	70.85
PointPillars + point-wise attention	90.53	88.75	87.25	88.84	50.18	46.40	43.62	46.73	84.49	69.75	65.53	73.25
PointPillars + SOPA	90.56	88.77	87.26	88.86	54.67	50.85	47.43	50.97	85.09	73.38	69.71	76.06

Table 4: Results on the KITTI test BEV detection benchmark.

Model	Speed (Hz)	mAP	Car			Pedestrian			Cyclist		
		Mod.	Easy (%)	Mod. (%)	Hard (%)	Easy (%)	Mod. (%)	Hard (%)	Easy (%)	Mod. (%)	Hard (%)
MV3D [36]	2.8	N/A	86.02	76.90	68.49	N/A	N/A	N/A	N/A	N/A	N/A
Cont-Fuse [37]	16.7	N/A	88.81	85.83	77.33	N/A	N/A	N/A	N/A	N/A	N/A
Roarnet [38]	10	N/A	88.20	79.41	70.02	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN [39]	10	64.11	88.53	83.79	77.90	58.75	51.05	47.54	68.09	57.48	50.77
F-PointNet [40]	5.9	65.39	88.70	84.00	75.33	58.09	50.22	47.20	75.38	61.96	54.68
HDNET [41]	20	N/A	89.14	86.57	78.32	N/A	N/A	N/A	N/A	N/A	N/A
PIXOR++ [41]	35	N/A	89.38	83.70	77.97	N/A	N/A	N/A	N/A	N/A	N/A
VoxelNet	4.4	58.25	89.35	79.26	77.39	46.13	40.74	38.11	66.70	54.76	50.55
SECOND	20	60.56	88.07	79.37	77.95	55.10	46.27	44.76	73.67	56.04	48.78
PointPillars	62	66.19	88.35	86.10	79.83	58.66	50.23	47.19	79.14	62.25	56.00
Ours	34	69.96	92.76	88.37	85.31	60.67	54.13	48.22	82.22	67.38	59.86

As shown in Tables 1–3, the ablation experimental results show that the accuracy of SOPA is overall improved by 1–2% mAP compared to adding point-wise attention. The points in the pillars are correlated with each other, thus processing the points in the point cloud individually will inevitably drop part of the useful geometric information, further affecting the detection accuracy. SOPA relates points within the same pillar to retain more meaningful information.

As indicated by our results in Tables 6–8, from the ablation experimental results, we can observe that the accuracy of adding the SOCA is superior to that of the only-fused channel attention mechanism, and the accuracy of adding the SOCA has overall improvement of 4–5% mAP compared with the existing method PointPillars. In particular, from Tables 5 and 9, we can see that the detection results of cyclist categories with slightly low detection accuracy are improved

Table 5: Results on the KITTI test 3D detection benchmark.

Model	Speed (Hz)	mAP Mod.	Car Easy (%)	Car Mod. (%)	Car Hard (%)	Pedestrian Easy (%)	Pedestrian Mod. (%)	Pedestrian Hard (%)	Cyclist Easy (%)	Cyclist Mod. (%)	Cyclist Hard (%)
MV3D [36]	2.8	N/A	71.09	62.35	55.12	N/A	N/A	N/A	N/A	N/A	N/A
Cont-Fuse [37]	16.7	N/A	82.54	66.22	64.04	N/A	N/A	N/A	N/A	N/A	N/A
Roarnet [38]	10	N/A	83.71	73.04	59.16	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN [39]	10	55.62	81.94	71.88	66.38	50.80	42.81	40.88	64.00	52.18	46.61
F-PointNet [40]	5.9	57.35	81.20	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39
VoxelNet	4.4	49.05	77.47	65.11	57.73	39.48	33.69	31.50	61.22	48.36	44.37
SECOND	20	56.69	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
PointPillars	62	59.20	79.05	74.99	68.30	52.08	43.53	41.49	75.78	59.07	52.92
Ours	34	63.08	87.48	76.22	73.55	54.78	49.45	43.07	81.69	63.58	60.55

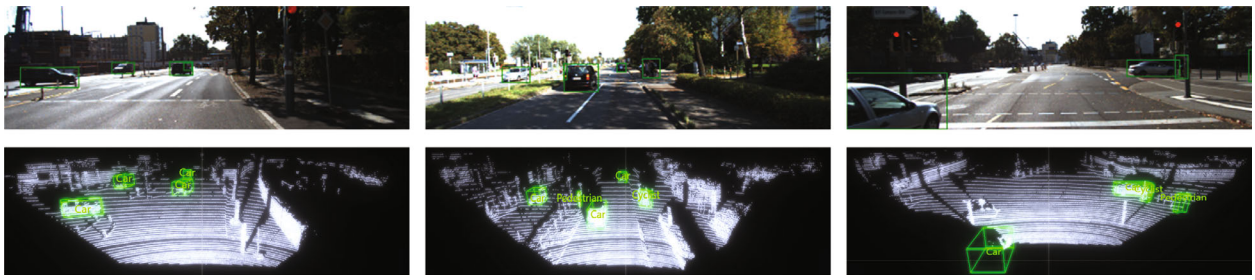


Figure 3: Visualization of our results. Qualitative results of the proposed method on the KITTI validation split. For each sample, the upper part is the image labeled with the predicted 3D bounding box, and the lower part is a representative view of the corresponding point clouds (best viewed when zoomed-in).

Table 6: Results on the KITTI test BEV detection benchmark.

Model	Car Easy (%)	Car Mod. (%)	Car Hard (%)	Car mAP (%)	Pedestrian Easy (%)	Pedestrian Mod. (%)	Pedestrian Hard (%)	Pedestrian mAP (%)	Cyclist Easy (%)	Cyclist Mod. (%)	Cyclist Hard (%)	Cyclist mAP (%)
PointPillars	88.35	86.10	79.83	84.76	58.66	50.23	47.19	52.02	79.14	62.25	56.00	65.79
PointPillars + channel attention	89.38	85.39	83.43	86.06	57.38	51.84	48.12	52.44	79.62	63.59	60.38	67.86
PointPillars + SOCA	89.50	86.30	83.80	86.53	58.37	53.07	49.18	53.54	81.26	64.75	61.87	69.29

Table 7: Results on the KITTI test 3D detection benchmark.

Model	Car Easy (%)	Car Mod. (%)	Car Hard (%)	Car mAP (%)	Pedestrian Easy (%)	Pedestrian Mod. (%)	Pedestrian Hard (%)	Pedestrian mAP (%)	Cyclist Easy (%)	Cyclist Mod. (%)	Cyclist Hard (%)	Cyclist mAP (%)
PointPillars	79.05	74.99	68.30	74.11	52.08	43.53	41.49	45.70	75.78	59.07	52.92	55.35
PointPillars + channel attention	82.68	75.37	72.10	76.71	52.19	45.82	42.48	46.83	75.44	59.03	55.61	63.36
PointPillars + SOCA	83.92	76.32	72.77	77.67	53.91	47.35	42.90	48.07	78.58	61.08	57.54	65.73

by a large margin (6% 3D mAP and 7% AOS mAP) on the detection benchmark of 3D and AOS. In the backbone network, each channel is processed separately in isolation. Ignoring the correlation between channels will lose some valuable information and decrease the detection precision. And SOCA associates channels with channels to retain more useful feature information.

The residual influencing factors might be the selection of various hyperparameters, including network design (convolution size, number of convolution layers, convolution type, and number of channels), projection using only a bird's eye view or incorporating a front view, whether to choose pitch angle as main parameter in the pillar feature net, choice of single or multiple detection heads, and lots more, which

Table 8: Results on the KITTI test average orientation similarity (AOS) detection benchmark.

Model	Car				Pedestrian				Cyclist			
	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)	Easy (%)	Mod. (%)	Hard (%)	mAP (%)
PointPillars	90.19	88.76	86.38	88.44	32.39	31.41	29.84	31.21	82.43	68.16	61.96	70.85
PointPillars + channel attention	90.55	88.75	87.09	88.79	54.61	50.27	47.38	50.75	84.24	70.09	65.78	73.37
PointPillars + SOCA	90.61	88.91	87.24	88.92	58.30	53.48	50.37	54.05	83.99	71.85	68.41	74.75

Table 9: Results on the KITTI test average orientation similarity (AOS) detection benchmark.

Model	Speed (Hz)	mAOS	Car			Pedestrian			Cyclist		
		Mod.	Easy (%)	Mod. (%)	Hard (%)	Easy (%)	Mod. (%)	Hard (%)	Easy (%)	Mod. (%)	Hard (%)
SubCNN [42]	0.5	72.71	90.61	88.43	78.63	78.33	66.28	61.37	71.39	63.41	56.34
AVOD-FPN [39]	10	63.19	89.95	87.13	79.74	53.36	44.92	43.77	67.61	57.53	54.16
SECOND	20	54.53	87.84	81.31	71.95	51.56	43.51	38.78	80.97	57.20	55.14
PointPillars	62	68.86	90.19	88.76	86.38	32.39	31.41	29.84	82.43	68.16	61.96
Ours	34	72.14	94.89	91.09	88.47	56.52	50.99	47.57	86.80	74.34	70.67

requires more experimental studies to separate and evaluate each potential influence factor.

7. Conclusions

This paper presents an object detection algorithm based on PointPillars by combining multiple attention mechanisms. A novel deep network and encoder, which improves the traditional end-to-end algorithm and adds multiple attention mechanisms to the network structure in the stage of feature extraction, improves the effectiveness of image feature extraction. On KITTI dataset, the algorithm provides higher detection performance (BEV, 3D and AOS mAP) at a relatively competitive speed. Our experimental results show that our point cloud object detection algorithm using multiple attention mechanisms is an excellent network for LiDAR 3D object detection at present, and the comparison with PointPillars further demonstrates the effectiveness of the proposed method in this paper. It is worth noting that the proposed object detection algorithm can be extended into intelligent transportation systems [43, 44] and private transmission systems [45, 46].

Data Availability

We use xy resolution: 0.16 m, maximum number of points per pillar (V): 100, maximum number of pillars (Z): 12000. Our approach is based on the pytorch framework, with all networks trained on the NVIDIA 2080ti computing platform.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Guangzhou Science and Technology Project under Grant 202102021132, in part by National Nature Science Foundation of China under Grant 62173101, in part by Guangzhou Key Laboratory of Software-Defined Low Latency Network under Grant 202102100006, in part by the Open Research Project of Zhijiang Laboratory under Grant 2021KF0AB06, and in part by the International Collaborative Research Program of Guangdong Science and Technology Department under Grants 2020A0505100061.

References

[1] A. Tizghadam, H. Khazaei, M. H. Moghaddam, and Y. Hassan, "Machine learning in transportation," *Journal of Advanced Transportation*, vol. 2019, Article ID 4359785, 3 pages, 2019.

[2] P. Salva-Garcia, J. M. Alcaraz-Calero, Q. Wang, J. B. Bernabe, and A. Skarmeta, "5G NB-IoT: efficient network traffic filtering for multitenant IoT cellular networks," *Security and Communication Networks*, vol. 2018, Article ID 9291506, 21 pages, 2018.

[3] M. Rasib, M. A. Butt, S. Khalid et al., "Are self-driving vehicles ready to launch? An insight into steering control in autonomous self-driving vehicles," *Mathematical Problems in Engineering*, vol. 2021, Article ID 6639169, 22 pages, 2021.

[4] Y. Zhou and O. Tuzel, "Voxelnet: end-to-end learning for point cloud based 3D object detection," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4490–4499, Salt Lake City, UT, USA, 2018.

[5] B. Yang, W. Luo, and R. Urtasun, "Pixor: real-time 3D object detection from point clouds," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7652–7660, Salt Lake City, UT, USA, 2018.

[6] M. Simony, S. Milzy, K. Amendey, and H. M. Gross, "Complex-yolo: an Euler-region-proposal for real-time 3D object

detection on point clouds," in *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pp. 197–209, Munich, Germany, 2018.

[7] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3D LiDAR using fully convolutional network," in *Proceedings of the Robotics: Science and Systems*, pp. 1–8, MI, USA, 2016.

[8] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: an efficient probabilistic 3D object detector for autonomous driving," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12677–12686, Long Beach, CA, USA, 2019.

[9] R. Qi, H. Su, K. C. Mo, and L. P. Guibas, "PointNet: deep learning on point sets for 3D classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 75–85, Honolulu, HI, USA, 2017.

[10] Y. Yan, Y. Mao, and B. Li, "SECOND: sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[11] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: fast encoders for object detection from point clouds," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12697–12705, Long Beach, CA, USA, 2019.

[12] Z. Liu, X. Zhao, T. Huang, R. Hu, Y. Zhou, and X. Bai, "Tanet: robust 3D object detection from point clouds with triple attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11677–11684, New York, USA, 2020.

[13] M. Corbetta and G. L. Shulman, "Control of goal-directed and stimulus-driven attention in the brain," *Nature Reviews Neuroscience*, vol. 3, no. 3, pp. 201–215, 2002.

[14] M. Hayhoe and D. Ballard, "Eye movements in natural behavior," *Trends in Cognitive Sciences*, vol. 9, no. 4, pp. 188–194, 2005.

[15] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, vol. 37, pp. 448–456, Lille, France, 2015.

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, Haifa, Israel, 2010.

[17] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference*, pp. 21–37, Amsterdam, The Netherlands, 2016.

[18] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Computer Vision – ECCV 2018*, pp. 784–799, Munich, Germany, 2018.

[19] D. Zhou, J. Fang, X. Song et al., "Iou loss for 2D/3D object detection," in *2019 International Conference on 3D Vision (3DV)*, pp. 85–94, Quebec City, QC, Canada, 2019.

[20] F. Wang and D. M. Tax, "Survey on the attention based RNN model and its applications in computer vision," in *Proceedings of the Computer Vision and Pattern Recognition*, pp. 1–42, Las Vegas, NV, USA, 2016.

[21] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Advances in neural information processing systems*, p. 28, Curran Associates, Inc, 2015.

[22] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "Cbam: convolutional block attention module," in *Computer Vision – ECCV 2018*, pp. 3–19, Munich, Germany, 2018.

[23] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-net: efficient channel attention for deep convolutional neural networks," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR),*, pp. 11531–11539, Seattle, WA, USA, 2020.

[24] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Advances in neural information processing systems*, p. 30, Curran Associates, Inc, 2017.

[25] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794–7803, Salt Lake City, UT, USA, 2018.

[26] X. Li, Z. Zhong, J. Wu, Y. Yang, Z. Lin, and H. Liu, "Expectation-maximization attention networks for semantic segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9167–9176, Seoul, Republic of Korea, 2019.

[27] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," in *Advances in neural information processing systems,*, p. 32, Curran Associates, Inc, 2019.

[28] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: criss-cross attention for semantic segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV),*, pp. 603–612, Seoul, Republic of Korea, 2019.

[29] L. Chen, H. Zhang, J. Xiao et al., "SCA-CNN: spatial and channel-wise attention in convolutional networks for image captioning," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6298–6306, Honolulu, HI, USA, 2017.

[30] X. Zhu, D. Cheng, Z. Zhang, S. Lin, and J. Dai, "An empirical study of spatial attention mechanisms in deep networks," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV),*, pp. 6688–6697, Seoul, Republic of Korea, 2019.

[31] H. Zhao, Y. Zhang, S. Liu et al., "PSANET: point-wise spatial attention network for scene parsing," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 267–283, Munich, Germany, 2018.

[32] H. Zhang, C. Wu, Z. Zhang et al., "Resnest: split-attention networks," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2735–2745,, New Orleans, LA, USA, 2022.

[33] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3354–3361, Providence, RI, USA, 2012.

[34] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2999–3007, Venice, Italy, 2017.

[35] D. P. Kingma and J. L. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference for Learning Representations, pp. 1-15, ICLR*, p. 13, San Diego, 2015.

[36] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6526–6534, Honolulu, HI, USA, 2017.

[37] M. Liang, B. Yang, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 641–656, Munich, Germany, 2018.

[38] K. Shin, Y. P. Kwon, and M. Tomizuka, "Roarnet: a robust 3D object detection based on region approximation refinement," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2510–2515, Paris, France, 2019.

[39] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, Madrid, Spain, 2018.

[40] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from rgb-d data," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 918–927, Salt Lake City, UT, USA, 2018.

[41] B. Yang, M. Liang, and R. Urtasun, "Hdnet: exploiting hd maps for 3D object detection," in *Conference on Robot Learning*, vol. 87, pp. 146–155, Zürich, Switzerland, 2018.

[42] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 924–933, Santa Rosa, CA, USA, 2017.

[43] Z. Ning, S. Sun, X. J. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, vol. 21, no. 12, pp. 4201–4217, 2022.

[44] X. Wang, Z. Ning, L. Guo, S. Guo, X. Gao, and G. Wang, "Mean-field learning for edge computing in mobile blockchain networks," *IEEE Transactions on Mobile Computing*, pp. 1–17, 2022.

[45] M. Wen, Q. Li, K. J. Kim et al., "Private 5G networks: concepts, architectures, and research landscape," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 1, pp. 7–25, 2022.

[46] M. Wen, E. Basar, Q. Li, B. Zheng, and M. Zhang, "Multiple-mode orthogonal frequency division multiplexing with index modulation," *IEEE Transactions on Communications*, vol. 65, no. 9, pp. 3892–3906, 2017.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9446418, 14 pages
https://doi.org/10.1155/2022/9446418

Research Article

A Dueling Deep Recurrent Q-Network Framework for Dynamic Multichannel Access in Heterogeneous Wireless Networks

Haitao Chen, Haitao Zhao, Li Zhou, Jiao Zhang, Yan Liu, Xiaoqian Pan, Xingguang Liu, and Jibo Wei

College of Electronic Science and Technology, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Haitao Zhao; haitaozhao@nudt.edu.cn

Received 15 July 2022; Accepted 31 August 2022; Published 1 October 2022

Academic Editor: Xiaojie Wang

This paper investigates a deep reinforcement learning algorithm based on dueling deep recurrent Q-network (Dueling DRQN) for dynamic multichannel access in heterogeneous wireless networks. Specifically, we consider the scenario that multiple heterogeneous users with different MAC protocols share multiple independent channels. The goal of the intelligent node is to learn a channel access strategy that achieves high throughput by making full use of the underutilized channels. Two key challenges for the intelligent node are (i) there is no prior knowledge of spectrum environment or the other nodes' behaviors; (ii) the spectrum environment is partially observable, and the spectrum states have complex temporal dynamics. In order to overcome the aforementioned challenges, we first embed the long short-term memory layer (LSTM) into the deep Q-network (DQN) to aggregate historical observations and capture the underlying temporal feature in the heterogeneous networks. And second, we employ the dueling architecture to overcome the observability problem of dynamic environment in neural networks. Simulation results show that our approach can learn the optimal access policy in various heterogeneous networks and outperforms the state-of-the-art policies.

1. Introduction

With the development of mobile communication [1], internet of things (IoT) [2], and virtual reality (VR) [3], the global mobile data traffic increases seven-fold between 2016 and 2021 with a compound annual growth rate of 46%, and it will reach 160 exabytes per month in 2025. Efficient spectrum utilization is crucial for future wireless networks to cater to the explosive growth of mobile data traffic. Dynamic spectrum access (DSA) is believed to be one of the key technologies to improve spectrum efficiency in the limited frequency bands [4]. However, most DSA technologies either rely on the prior information of the network or cannot effectively adapt to the real wireless network environment with complex and dynamic features. Motivated by these considerations, we propose a deep reinforcement learning (DRL) framework for dynamic multichannel access in heterogeneous wireless networks.

At the first step, we consider the efficient spectrum utilization in a multichannel heterogeneous wireless network. On one hand, with the development of network technology and the opening of unlicensed frequency bands, the next-generation wireless networks will become more heterogeneous and complex due to the types of emerging radio access networks (RANs) they integrate and the types of numerous applications they support. On the other hand, in order to further improve the capacity and performance of the wireless network, multiband aggregation will be a significant trend in the next-generation network deployment. Based on these premises, the first goal of this paper is to investigate a clean-slate design in which the intelligent node shares the spectrum with these nodes from different networks in a dynamic way. There is no cooperation mechanism and information exchange between the intelligent node and other nodes. The intelligent node needs to learn the optimal channel access policy by a chain of observations and actions

without any prior knowledge about the system statistics (e.g., the MAC mechanisms of other nodes and the state transition probability of each channel).

Since the intelligent node can only obtain the state of channels chosen by itself, the whole spectrum environment is partially observable. The multichannel access problem is a partially observable Markov decision process (POMDP) for the intelligent node. In theory, POMDP is polynomial-space-hard (PSPACE-hard) [5], and the increase in the dimension of states will result in double-exponential growth in complexity. Hence, the traditional DRL method, e.g., the deep Q-network (DQN), may be unable to find an optimal access policy. Thus, in this paper, the long short-term memory [6], a powerful variant of the recurrent neural network (RNN), is embedded into the DQN to capture the related state information from the historical observations and learn the dynamics of the spectrum environment. Further, considering the observability problem about the dynamic environment in DSA, we introduce the dueling architecture [7] in our neural network, which can determine the best action for each state in the large dynamic unknown environments. We refer to our proposed DRL algorithm as Dueling DRQN.

The main contributions in this work can be summarized as follows:

(i) We propose a DRL-based framework for dynamic multichannel access in heterogeneous wireless networks. In contrast to traditional DSA technologies, our scheme does not rely on any prior information of the network and can intelligently select underutilized channels by itself in each time slot

(ii) We introduce the LSTM into our neural network to overcome the partially observation problem about the spectrum environment. In addition, we add the dueling architecture in our neural network to evaluate comprehensive states and make the best decision

(iii) For the sake of avoiding Dueling DRQN falling into the local optimum, we propose an adaptive ε-greedy method to balance exploitation and exploration

(iv) Extensive simulation results demonstrate that the proposed Dueling DRQN algorithm can achieve higher throughput performance and lower collision rate compared with the baseline algorithms. Moreover, our scheme performs good adaptiveness when the spectrum environment dynamically changes over time

1.1. Related Work. Over the past decade, a variety of methods have been investigated for the DSA problem in wireless networks [8–16]. They are divided into two mainstreams in general: model-dependent methods and model-free methods. For model-dependent methods, it relies on establishing the accurate system model, based on which an effective access policy can be acquired. The myopic policy [8] and whittle index policy [9] are two classic model-dependent methods. When channels are independent and identically distributed (i.i.d.), the myopic policy performs well. But the myopic policy does not have any performance guarantee when channels are correlated or follow different distributions. The whittle index policy has similar problem. According to [9], when all channels are independent and the two-state Markov chain transition matrix is known, the whittle index policy can be regarded as a close-form solution. When all channels are identical, the whittle index policy has the same performance as the myopic policy. However, if there are some channels that cannot be modelled in the form of two-state Markov transition matrices, the performance of these two model-dependent methods will deteriorate.

Instead, for model-free methods, the optimal policy is acquired through interacting with the environment. The reinforcement learning (RL), a typical representative of model-free methods, is widely applied to investigate the DSA problems [12–16]. The most popular algorithm is Q-learning since it is easy to implement and has better performance than traditional algorithms. However, Q-learning needs to maintain a tabulation to store and update the Q-value during the interaction with the environment. When the state space or action space becomes larger, or the environment is partially observable, the complexity of tabulating the Q-values is fairly high, and the performance of Q-learning becomes inefficient. Fortunately, by introducing the deep neural network in RL, DRL can deal with these problems with very large state and action spaces, which has attracted more and more attention in recent years [17, 18].

Many existing works have adopted DRL to solve the DSA problems. For example, authors in [19] examined the correlated channels in DSA, which is a pioneer work using the DRL for the channel access. In [20], a deep actor-critic reinforcement learning based framework for dynamic multichannel access was proposed in both the single-user and multiple-user scenarios. The simulation results showed that the proposed framework had good performance in handling a large number of channels. In addition, a methodology that utilized DRL to implement a fair multichannel access strategy in a distributed wireless network (DWN) was developed in [21]. The authors removed the assumption that all nodes in DWN are in the saturated mode, which is closer to the practical scenarios. In [22–25], the authors concentrated on the DRL-based DSA strategies in cognitive radio network (CRN). However, all the network scenarios considered in these works are homogeneous. Unlike [19–25], in this paper, we pay more attention to the coexistence of nodes adopting various MAC protocols in the shared multichannel heterogeneous network. The spectrum environment is more complicated than these works. In addition, our scheme needs to keep flexible and efficient spectrum access strategies so that it can live in harmony with other nodes.

There are a few works that investigate the DSA problem in heterogeneous networks, such as [26–29]. However, the authors in [26, 27] only considered a single channel that is shared by the intelligent node and other nodes. The intelligent node only needs to decide whether transmit or not, which greatly reduces the difficulty of the DSA problem. Both [28, 29] investigated the joint user association and

spectrum allocation issue in multichannel heterogeneous networks and proposed the multiagent DRL schemes to solve the computationally expensive optimization problem with the large state and action space. However, they hold an assumption that each user can obtain the global state space about other users in the network and ignore the partial observability about external environment. Actually, real-world tasks often feature incomplete state information caused by partial observability [30]. It is extremely difficult to collect the global state information because of the large communication overhead and processing cost. While in this paper, our proposed scheme does not rely on the global state information, and the intelligent node strives to achieve optimal performance based on limited information about spectrum environment. Table 1 summarizes the differences of our proposed scheme, comparing to some existing works that previously reported.

The rest of the paper is organized as follows. Section 2 introduces the system model and formulates the DSA problem by POMDP. Section 3 proposes the Dueling DRQN algorithm and explains it in detail. Simulation results and performance analyses are presented in Section 4. Finally, we conclude this paper in Section 5.

2. System Model and Problem Formulation

2.1. System Model. As shown in Figure 1, the system model we consider in this paper is a time-slotted multichannel (including N orthogonal channels) heterogeneous wireless network where multiple radio networks coexist with each other. Each radio network is independent of each other, and the channels for different networks have been preallocated by AP, which means that different radio networks do not interfere with each other. We assume that all the nodes are backlogged, i.e., they always have packets to transmit. A node can transmit at the beginning of a time slot and must finish transmission within that time slot. For the AP, it can receive data from different radio networks on different channels in each time slot simultaneously. Thereafter, AP can broadcast corresponding ACK information on different channels to indicate whether the transmissions are successful or not.

The entire network is heterogeneous because different radio networks may adopt different time-slotted MAC protocols. Specifically, we assume that each node chooses one of N orthogonal channels at each time slot to transmit its data packets. The types of nodes and their transmission manners are described below:

(i) *Authorized Node*. Occupies a fixed channel and transmits all the time

(ii) *TDMA Node*. Occupies a fixed channel, but transmits in X specific time slots within each frame of Y time slots in a repetitive manner from frame to frame

(iii) *Hopping Node*. Dynamically occupies several channels which change at each time slot following a fixed regular pattern

(iv) *q -ALOHA Node*. Occupies a fixed channel and transmits with a probability q in each time slot in an i.i.d. manner

(v) *Stochastic Node*. Occupies a fixed channel based on a probability distribution such as a two-state Markov chain

As depicted in Figure 2, numerous nodes belonging to different radio networks and their own transmission manners jointly lead to a complex and dynamic spectrum environment. Meanwhile, we can find there still exist many idle spectrums that are not fully utilized due to the fixed channel preallocation method. An intuitive motivation of this paper is how to improve the spectral efficiency of the network on the premise of overcoming the complex and dynamic external spectrum environment. Specifically, the DRL node is an intelligent node that implements our proposed Dueling DRQN algorithm. The DRL node needs to adopt an appropriate access strategy to make use of the idle spectrum and avoid collisions with other nodes.

However, there are two key challenges for the DRL node. First, the prior system information, such as the MAC mechanism of different nodes and the channel state information, is not known by the DRL node. Second, the DRL node can only obtain the state of the selected channel through the ACK signal from the AP. In other words, the whole channels' states are partially observable from the perspective of the DRL node. The above challenges make these classical approaches for DSA ineffective and even difficult to implement. Thus, a new online learning method is needed.

2.2. Problem Formulation. Because the prior system statistics are unknown and the full channel state information is partially observable by the DRL node, we use the POMDP to model the dynamic multichannel access process, which is described by the action, state, reward, and state transition function.

2.2.1. Action. The action of the DRL node is to select which channel to access in each time slot, and all possible actions constitute the action space $\mathscr{A}$, $a_t \in \mathscr{A}$. The action is denoted as a_t at time slot t, $a_t = \{a_{1,t}, a_{2,t}, \cdots, a_{N,t}\}$, where

$$a_{i,t} = \begin{cases} 1, & \text{if the DRL node chooses channel } i \text{ to access,} \\ 0, & \text{other.} \end{cases} \tag{1}$$

2.2.2. State. Due to the different transmission strategies of these existing nodes, the channel states will change at each time slot. We define the full channel states at time slot t as Ω_t, which is

$$\Omega_t = \{\omega_{1,t}, \omega_{2,t} \cdots, \omega_{i,t}, \cdots, \omega_{N,t}\}, 1 \le i \le N, \tag{2}$$

where $\omega_{i,t}$ is the binary representation of the state of channel i: occupied (1) or idle (-1). After taking action a_t, the DRL

Table 1: Differences between the proposed scheme and existing works in DSA problem.

	Network scenarios	Model-dependent or model-free	Multichannel access	Partial observability	Time-varying environment
[8–11]	Homogeneous networks	Model-dependent	Yes	No	No
[12–16]	Homogeneous networks	Model-free	Yes	No	No
[19]	Homogeneous networks	Model-free	Yes	Yes	No
[20]	Homogeneous networks	Model-free	Yes	Yes	Yes
[21]	Heterogeneous networks	Model-free	No	Yes	No
[22, 23, 25]	Homogeneous networks	Model-free	Yes	No	No
[24]	Homogeneous networks	Model-free	Yes	Yes	No
[26–27]	Heterogeneous networks	Model-free	No	No	No
[28–29]	Heterogeneous networks	Model-free	Yes	No	No
Ours	Heterogeneous networks	Model-free	Yes	Yes	Yes

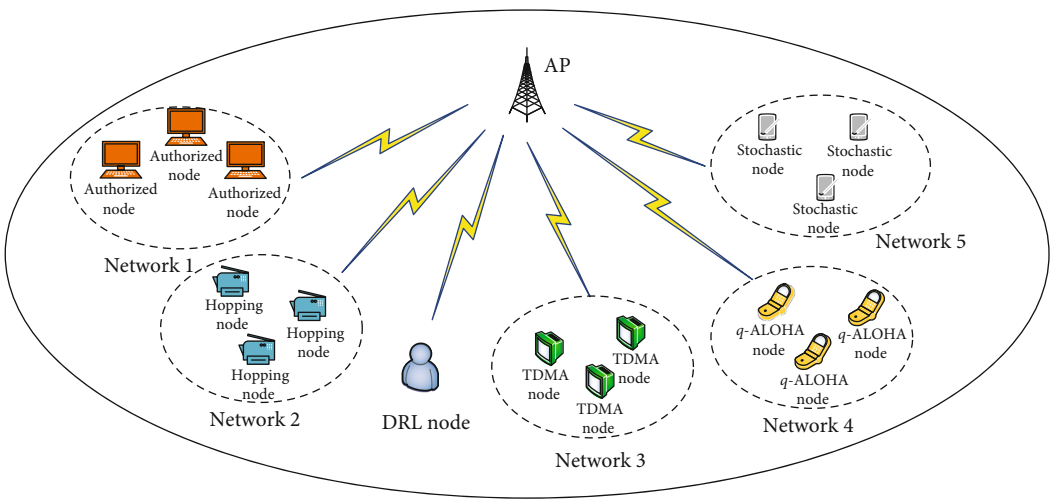


Figure 1: The architecture of heterogeneous wireless networks.

node gets an observation $o_t = \{o_{1,t}, o_{2,t} \cdots, o_{i,t}, \cdots, o_{N,t}\}$. If the DRL node chooses channel i to access,

$$o_{i,t} = \begin{cases} 1, & \text{transmission is successful} \\ -1, & \text{transmission is collided} \end{cases} \tag{3}$$

for $j \neq i$, $o_{j,t} = 0$, which represents the state of channel j is unknown by the DRL node. The state of the DRL node at time slot $t+1$ is defined as the past l -length action-observation pairs up to the time slot t, i.e.,

$$s_{t+1} = (a_{t-l+1}, o_{t-l+1}, a_{t-l+2}, o_{t-l+2}, \cdots a_t, o_t). \tag{4}$$

2.2.3. Reward. After performing the action a_t, the transition from state s_t to s_{t+1} generates a reward r_{t+1}. The immediate reward r_{t+1} is based on the successful transmission or conflicting transmission. Specifically, the reward r_{t+1} is defined as

$$r_{t+1} = \begin{cases} 1, & \text{if } o_{i,t} = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Note that the object of our proposed scheme is to improve the spectrum utilization of the network while avoiding collisions. However, the spectrum efficiency is related to the number of successful transmissions directly.

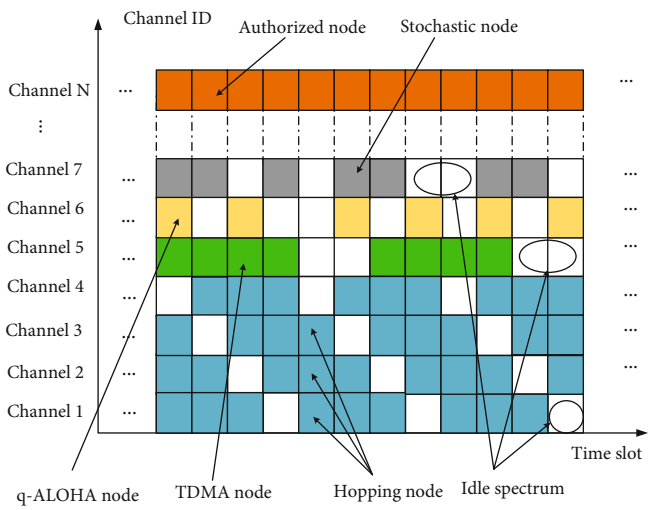


Figure 2: Complex and dynamic spectrum environment.

Thus, we pay more attention to the successful transmission in the following sections of this paper. In addition, the transmission is failed if the collision occurs.

2.2.4. State Transition Function. At time slot t, the state transition function is denoted as ψ_t, and ψ_t is the probability that the DRL node executes action a_t at the state s_t and moves to the next state s_{t+1}. As interaction times increase, ψ_t will gradually converge to the optimal ψ_t^* by maximizing the long-term accumulated discounted reward, i.e.,

$$\psi_t^* \longleftarrow \max \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \tag{6}$$

where $0 \le \gamma \le 1$ is a discounted factor indicating the important degree of future reward to the current reward.

3. The Proposed Dueling DRQN

The framework of Dueling DRQN includes experience replay, deep neural networks, and reinforcement learning, which is shown in Figure 3.

3.1. Experience Replay. During the learning process, the experience-replay pool $\mathscr{D}$ [31] stores the current state, action, reward, and the next state as a tuple $(s_t, a_t, r_{t+1}, s_{t+1})$, which is referred to as experience E_t at time slot t. The experience-replay pool $\mathscr{D}$ accumulates the history experiences for many time slots. When training the Q-network, the network is trained by a minibatches M_E of experiences that are sampled from $\mathscr{D}$ randomly, rather than only using current experience. Experience replay can increase data efficiency through the reuse of experience samples. More importantly, it reduces the correlation among the training data by randomly sampling data in experience-replay pool.

3.2. Deep Neural Networks. The mapping from states to actions follows the policy π, i.e., $a_t \sim \pi(s_t)$. The performance of policy π is evaluated by the state-action value function $Q^\pi(s, a)$, which is denoted as the expected accumulated discount reward for executing action a at the state s following the policy π, i.e.,

$$\begin{aligned} \mathscr{R}_t &= \sum_{t=0}^{\infty} \gamma^t r_{t+1}, \\ Q^\pi(s, a) &= \mathbb{E}[\mathscr{R}_t | s_t = s, a_t = a, \pi]. \end{aligned} \tag{7}$$

Moreover, deep neural networks are used to fit the state-action value function by $Q^\pi(s, a; \theta)$, rather than computing and storing the true $Q^\pi(s, a)$ in Q-table, i.e., $Q^\pi(s, a; \theta) \approx Q^\pi(s, a)$. The parameter θ is the set of weights in the deep neural networks. The estimation network and the target network have the same network structure but different hyperparameters.

Generally, deep neural networks have the strong nonlinear mapping and fitting ability. The reason is that the networks contain multiple hidden layers. However, for traditional deep neural networks (e.g., FNN), the output of each hidden layer depends only on the current input. In order to accommodate the partial observation, we should make full utilization of historical observations and infer the true channel state based on the historical experiences. Therefore, we improve the deep neural networks in our proposed algorithm by adding an LSTM layer and the dueling architecture. Figure 4 shows the detailed structure of deep neural networks in our proposed algorithm.

3.2.1. Long Short-Term Memory Network (LSTM). LSTM is a variant of the recurrent neural network, which overcomes the gradient exploding problem or the gradient vanishing problem in RNNs by introducing a gating mechanism. The LSTM layer maintains an intermediate state and aggregates observations over time, which is capable of capturing the information about historical observations. Therefore, we add an LSTM layer to estimate the true channel states and learn the dynamic pattern of the networks.

In LSTM, the intermediate state C_t at time slot t is used to retain historical information about channel states from initial time slot to time slot t. For each time slot t, C_t is determined by both the forget gate F_t and input gate I_t. First, the forget gate determines how much information stored in C_{t-1} needs to be forgotten, which is given by

$$F_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right), \tag{8}$$

where $\sigma = 1/(1 + e^{-x})$ is the logistic function [32]. x_t is the current input matrix. h_{t-1} is the hidden state at time slot $t-1$, which can be regarded as the short-term memory. W_f is the weight matrices of F_t. b_f is the bias term of F_t. Second, the input gate I_t controls how much information stored in the candidate state $\tilde{C}_t$ should be retained to update C_t. We calculate the input gate values I_t and the candidate state $\tilde{C}_t$ as

$$I_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i), \tag{9}$$

$$\tilde{C}_t = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right), \tag{10}$$

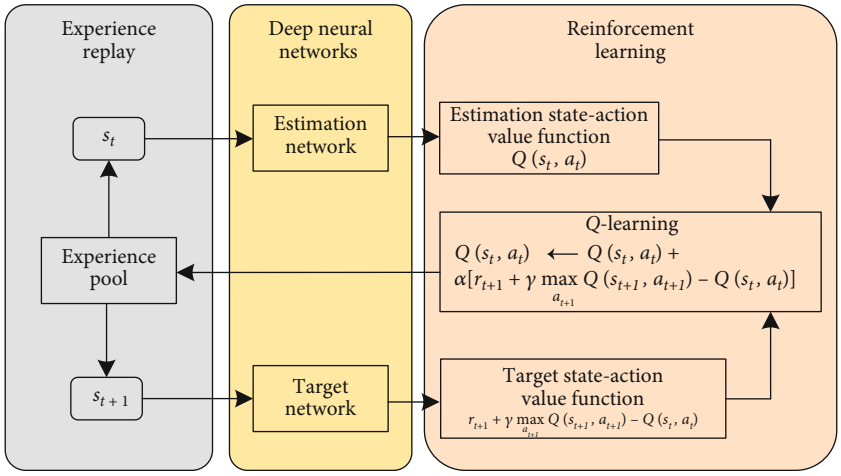


Figure 3: The framework of Dueling DRQN.

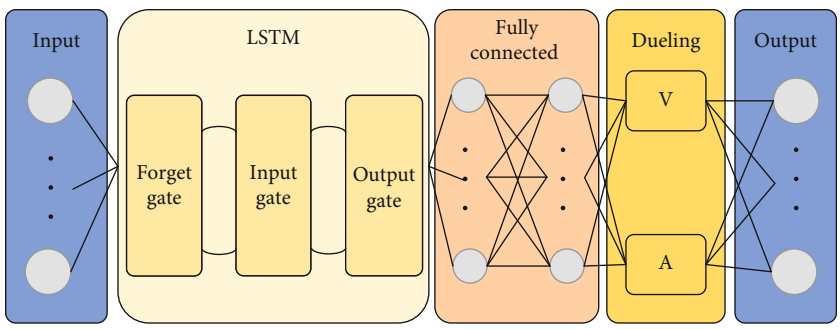


Figure 4: The structure of deep neural networks for Dueling DRQN algorithm.

where W_i and W_c are the weight matrices of I_t and $\tilde{C}_t$, respectively. b_i and b_c are the bias terms of I_t and $\tilde{C}_t$, respectively. $\tanh = (e^x - e^{-x})/(e^x + e^{-x})$ denotes the hyperbolic tangent function [33]. Given the values of F_t and I_t, the intermediate state C_t can be updated as

$$C_t = F_t \odot C_{t-1} + I_t \odot \tilde{C}_t, \tag{11}$$

where the symbol $\odot$ is the element multiplication.

The output gate O_t is used to control the amount of the information in C_t exported to the hidden state h_t at time slot t. O_t and h_t can be obtained as

$$O_t = \sigma(W_o[h_{t-1}, x_t] + b_o), \tag{12}$$

$$h_t = O_t \odot \tanh(C_t), \tag{13}$$

where W_o is the weight of O_t, and b_o is the bias of O_t.

In the LSTM model, C_t can capture the key information from x_t at each time slot and save it for a certain time interval. Based on C_t, the DRL node can predict the true channel state well. Finally, we can get the output of the LSTM as

$$y_t = W_y \cdot C_t + b_y, \tag{14}$$

where W_y and b_y are the output weights and the output bias, respectively.

3.2.2. Dueling Architecture. Factually, there is an observability problem about dynamic environment in the neural network. The state may be good or bad regardless of the taken action when there are multiple idle channels. Therefore, it is desirable to estimate the state-action value function $Q(s_t, a_t; \theta)$ by the value of state (i.e., $V(s_t; \theta)$) and the advantage of each action (i.e., $A(s_t, a_t; \theta)$) [7], which can be expressed as

$$Q(s_t, a_t; \theta) = V(s_t; \theta) + A(s_t, a_t; \theta), \tag{15}$$

where $V(s_t; \theta)$ estimates the expected value of state with respect to the taken action, and $A(s_t, a_t; \theta)$ denotes the advantage of each action minus the average value of the state with respect to the taken actions, i.e., $A(s_t, a_t; \theta) - (1/|\mathscr{A}|) \sum_{a' \in \mathscr{A}} A(s_t, a'; \theta)$. Copying the output streams from the fully connected layer and leading them to the value layer (V layer) and the advantage layer (A layer), we will get the expected value of the state by V layer and the comparative advantage of each action by A layer, which can evaluate the state-action value function more precisely and reduce redundant operations in action sampling.

Through the LSTM and dueling architecture, the estimation network approximates the estimation state-action value function eval_$Q = Q^{\pi}(s_t, a_t; \theta)$ with a set of weights θ, and the target network approximates a target state-action value functiontarget_$Q = r_{t+1} + \gamma \max_{a_{t+1} \in \mathscr{A}} Q^{\pi}(s_{t+1}, a_{t+1}; \theta^{-})$ with a set of weights θ^{-}. The loss function of Dueling DRQN is given as

$$L(\theta) = \mathbb{E}_{s_t, a_t, r_t, s_{t+1}} \left[(\text{target\_}Q\text{-eval\_}Q)^2 \right]. \tag{16}$$

The parameter of the estimation network θ is updated at every time slot t by

$$\theta \longleftarrow \theta + \alpha[\text{target\_}Q\text{-eval\_}Q] \nabla_{\theta} \text{eval\_}Q, \tag{17}$$

where $\alpha \in (0, 1]$ is the learning rate. ∇_{θ} denotes the gradient of θ. Finally, the parameter of the target network θ^{-} is updated to the parameter of the estimation network θ every K training times.

3.3. Q-Learning. Through the deep neural networks mentioned above, the output layer generates the state-action value functions corresponding to different actions, i.e., $Q^{\pi}(s_t, \forall a_t \in \mathscr{A}; ; \theta)$. For the sake of avoiding the DRL node falling into local optimal solution, it is necessary to trade off the exploitation (using the best known action) and the exploration (learning new, possibly better actions). The $\varepsilon -$ greedy policy is often used to keep this balance, which is described as

$$a_t = \begin{cases} \arg\max_{\tilde{a} \in \mathscr{A}} Q^{\pi}(s_t, \tilde{a}), & \text{with probability } 1\text{-}\varepsilon, \\ \text{random action}, & \text{with probability } \varepsilon, \end{cases} \tag{18}$$

where the DRL node selects an exploitation action with probability $1 - \varepsilon$ and selects an exploration action with probability ε.

The exploration probability is a key hyperparameter that directly affects the DRL node's decision. If ε is a small value, the DRL node becomes conservative, and it prefers to select the best known action based on the state-action value function. On the contrary, if ε is a large value, the DRL node becomes aggressive, and it prefers to select the new and unknown action. In the initial stage of algorithm implementation, due to the large state and action space, the DRL node should explore different possible states and actions frequently to enrich its knowledge about the environment. However, with the increasing number of iterations, the exploitation probability should increase accordingly so that the DRL node can make the best decision. Therefore, we design an adaptive $\varepsilon -$ greedy policy, which is expressed as

$$\varepsilon = \varepsilon_{\min} + (\varepsilon_{\max} - \varepsilon_{\min}) e^{-\varsigma t}, \tag{19}$$

where $\varepsilon_{\max}$ and $\varepsilon_{\min}$ are the maximal value and the minimal value of ε, respectively. ς is the decay factor.

After acquiring the action a_t by the adaptive $\varepsilon -$ greedy policy, the DRL node obtains the corresponding reward r_{t+1} and the new state s_{t+1} from the environment. The Q -learning will update $Q^{\pi}(s_t, a_t; \theta)$ following Bellman equation

$$\begin{aligned} Q^{\pi}(s_t, a_t; \theta) &\longleftarrow Q^{\pi}(s_t, a_t; \theta) \\ &+ \alpha \left[r_{t+1} + \gamma \max_{a_{t+1} \in \mathscr{A}} Q^{\pi}(s_{t+1}, a_{t+1}; \theta^{-}) - Q^{\pi}(s_t, a_t; \theta) \right]. \end{aligned} \tag{20}$$

With the training times increase, the state-action value function gradually converges to the optimal, and the policy π approaches the best policy π^* by maximizing the long-term accumulated discounted reward

$$\pi^* \longleftarrow \max \sum_{\tau = \tau_0}^{t} Q^{\pi^*}(s_{\tau}, a_{\tau}; \theta). \tag{21}$$

3.4. Dueling DRQN Training. The pseudocode of Dueling DRQN is given in Algorithm 1. To be specific, the training process is given as follows. Step 1: input the current state s_t into the estimation network and obtain the state-action value function for different actions. Step 2: choose an action a_t according to the adaptive $\varepsilon -$ greedy policy. Step 3: adopt this action to obtain observation o_t and the reward r_{t+1}. Step 4: generate the next state s_{t+1} and store the experience tuple $(s_t, a_t, r_{t+1}, s_{t+1})$ into experience pool $\mathscr{D}$. Step 5: compute the target_Q and train the estimation network. The hyperparameters of the estimation network are updated in real time. Step 6: update the hyperparameters of the target network every K time slots.

4. Experimental Results and Discussion

In this section, we investigate and analyze the performance of the proposed Dueling DRQN algorithm through simulations with Tersorflow1.19.0 on Python3.7 platform. We first describe the simulation settings, which are summarized in Table 2. After that, we investigate the coexistence of our proposed access scheme with other protocols mentioned above.

4.1. Simulation Setting

(1) *Hyperparameter Settings.* As shown in Figure 4, the number of neurons in LSTM layer and the fully connected layer are 128. The activation functions used for the neurons are ReLU functions [34], and the Adam algorithm [35] is used to conduct a stochastic gradient descent for the update of θ. The state history length l is 16. And the learning rate α of the DRL node is set to 0.001. The experience-replay pool size is set to1000, and the experience-replay pool is updated in a first-input-first-output manner: the oldest experience will be replaced by the new experience in it. The minibatch size M_E is set to 64. More detailed parameter settings are listed in Table 2

(2) *Performance Metric.* The main performance metric we consider in the paper is "throughput," which is

Initialize s_0,α,γ,$\varepsilon_{\max}$,$\varepsilon_{\min}$,ς,K,l,N.
Initialize experience pool $\mathscr{D}$ and mini-batches M_{E}.
Initialize the parameter of the estimation network as θ.
Initialize the parameter of the target network $\theta^{-} = \theta$.
For episode $i = 0, 1, \cdots, I$ **do**
  For time-slot $t = 0, 1, \cdots, T$ **do**
    Input s_t into the estimation network and output the $\{Q^{\pi}(s_t, a\,;\theta) | a \in \mathscr{A}\}$;
    Select the action a_t using the adaptive $\varepsilon - greedy$policy algorithm
      And update ε according to the equation (19);
    Execute action a_t and generate the observation o_t and r_{t+1};
    Compute s_{t+1} from s_t,a_t;
    Store$(s_t, a_t, r_{t+1}, s_{t+1})$ into the experience-replay pool $\mathscr{D}$.
    If$t \geq M_{\mathrm{E}}$**then**
      Randomly generate an index subset Θ;
      Sample $M_{\mathrm{E}}\{(s_j, a_j, r_{j+1}, s_{j+1})\}_{j\in\Theta}$ from $\mathscr{D}$;
      For each sample $(s_j, a_j, r_{j+1}, s_{j+1})$ in M_{E}**do**
        Compute the target_$Q = r_{j+1} + \gamma \max_{a_{j+1}\in\mathscr{A}} Q^{\pi}(s_{j+1}, a_{j+1}\,;\theta^{-})$ and obtain a_{j+1}.
      End for
      Calculate the loss function according to the equation (16) and update θ according to the equation (17);
      Minimize the loss function with learning rate α.
    End if
    Every K time slots: Update θ^{-} by setting $\theta^{-} = \theta$.
  End for
End for

Algorithm 1: Training process of Dueling DRQN.

Table 2: Parameter settings.

Parameter	Values
Episodes	20
The number of time slots in one episode	5500
State history length (l)	16
Experience-replay pool size	1000
Experience-replay minibatch size	64
Discount factor γ	0.9
Learning rate α	0.001
The maximal exploration probability $\varepsilon_{\max}$	0.8
The minimal exploration probability $\varepsilon_{\min}$	0.001
The decay factor ς	0.001
Target network update frequency K	100

defined as the probability of successful transmission for each episode, i.e., $T_{\tau} = \sum_{\tau=t-L+1}^{t} r_{\tau}/L$,where L is the number of time slots in one episode

(3) *Baselines*. To evaluate our proposed access policy, we consider the following baselines in this paper

(i) *DQN Access Policy*. The node accesses one channel in each time slot using the standard DQN [19]

(ii) *Whittle Index Policy*. The multichannel access strategy proposed in [9]

(iii) *Random Access Policy*. The node randomly accesses one channel in each time slot

(iv) *Optimal Access Policy*. The node is the model-aware node, and it has all the prior knowledge about the heterogeneous networks and the MAC mechanisms of the coexisting nodes, so it can access the channels using the optimal scheme

4.2. Simple Heterogeneous Network Scenarios. In this section, we investigate the performance of the proposed algorithm in four different simple heterogeneous network scenarios. Specifically, we set the number of channels is four, i.e., $N = 4$. Every scenario includes one authorized node and three other nodes which operate the same MAC protocol (there are only two hopping nodes in case II, and the details will be described in the following). The descriptions of four scenarios and related settings are summarized as follows:

(1) *Case I*. Coexistence with authorized node and TDMA node: in this case, we consider the coexistence of the DRL node with other protocol-based nodes including one authorized node and three TDMA nodes. The authorized node occupies channel 1 and transmits at all times. Three TDMA nodes occupy the rest three channels, respectively. The TDMA frame Y is set to 10, and the transmission slots X assigned to each TDMA node are slot 8, 5, and 2. It is noted that each TDMA node occupies

the according channel for consecutive X time slots and idle for the rest $Y - X$ time slots in a frame

(2) *Case II*. Coexistence with authorized node and hopping nodes: in this case, the DRL node coexists with one authorized node and two hopping nodes. The authorized node occupies channel 1 and transmits at all times. Two hopping nodes occupy the rest three channels, and the dynamic transmission pattern of the hopping nodes is $C2C3 \longrightarrow C3C4 \longrightarrow C4C2 \longrightarrow C2C3$

(3) *Case III*. Coexistence with authorized node and q-ALOHA nodes: in this case, the DRL node coexists with one authorized node and three q-ALOHA nodes. The authorized node occupies channel one and transmits at all times. Three q-ALOHA nodes occupy the rest three channels, and the transmission probability of each q-ALOHA node in different channels is [0.6, 0.9, 0.3]

(4) *Case IV*. Coexistence with authorized node and two-state Markov nodes: in this case, the DRL node coexists with one authorized node and three two-state Markov nodes. The authorized node occupies channel 1 and transmits at all times. The three two-state Markov nodes occupy the rest channels, and the transmission transition probability of these nodes on the n-th channel follows a two-state Markov chain

$$\mathscr{P}_n = \begin{pmatrix} p_{00}^n & p_{01}^n \\ p_{10}^n & p_{11}^n \end{pmatrix}, \tag{22}$$

where $p_{ij} = \Pr(\text{state}_j | \text{state}_i)$, and $\Pr(\cdot)$ denotes the transition probability. state_i represents the channel state of last time slot, and state_j represents the channel state of current time slot.

Figure 5(a) shows the experiment result of the coexistence with authorized node and TDMA nodes. As we can see, both the Dueling DRQN and DQN can quickly learn the variation characteristics of channel state and capture the idle channel without the transmission schedule of the TDMA nodes. Compared to the whittle index policy and random access policy, the throughput growths of the Dueling DRQN are more than 30% and 40%, respectively. In the whittle index policy, the node estimates the state transition probability matrices of all channels and selects one channel to access. However, because the frame length and the transmission time slots of the TDMA nodes are stationary, the state transition model of the TDMA channel cannot be captured by the two-state Markov transition model in the whittle index policy. This is why although the whittle index policy has the better performance than the random access policy, it is unable to make full utilization of the idle channels in this scenario. In addition, since all time slots are aligned and the transmission schedule of the TDMA nodes is fixed, we can calculate that the highest throughput achieved by the optimal access policy is $1 - \min(2/10, 5/10, 8/10) = 0.8$ in each episode.

Figure 5(b) presents the result of the coexistence with authorized node and hopping nodes. Even though the Dueling DRQN and the DQN achieve the approximate throughput, the Dueling DRQN has a faster convergence speed and smaller variance than the DQN during the convergence stage. For one thing, the idle slots are sparsely distributed in an episode, and there is only one idle channel in each time slot. In addition, the channel with idle slot switches according to the round-robin scheduling [14], which increases the difficulty of learning and convergence of DQN. For another thing, the Dueling DRQN can aggregate and make full utilization of historical data to predict the idle channel in the next time slot. Therefore, it is more efficient than the DQN. The Dueling DRQN outperforms the whittle index policy and the random access policy with the throughput improvement of 60% and 65%, respectively. In this scenario, the whittle index policy also performs poorly since the state transition process of all channels does not follow a two-state Markov transition model. Besides, the sparsity of underutilized channels further degrades the whittle index policy. Since there is always an idle channel in each time slot, so the optimal throughput in theory is one.

Figure 5(c) gives the result of the coexistence with authorized node and q-ALOHA nodes, and Figure 5(d) gives the result of the coexistence with authorized node and two-Markov nodes. Compared with case I and case II, the spectrum states in case III and case IV are more stochastic. In addition, the channel states have the temporal correlation in case IV. As we can see, the Dueling DRQN can reach convergence during $4 \sim 5$ episodes, but the DQN reaches the convergence during $8 \sim 10$ episodes. The reason is that the LSTM layer not only has the memory ability but also has the ability to infer the temporal correlation of channel states. However, the neural networks in standard DQN are fully connected layers that do not have the ability to memorize and ratiocinate. Therefore, the learning process of the Dueling DRQN is quicker than the DQN. In addition, compared to the whittle index policy and the random access policy, the throughput improvement of the Dueling DRQN is more than 30% and 45% in case III, respectively. However, in case IV, the two-state Markov transition model in the whittle index can capture the behavior of the two-Markov nodes absolutely. Thus, it can make full utilization of the underutilized channels and achieve the near-optimal performance. It is worth noting that the Dueling DRQN can also obtain a performance similar to the whittle index policy after a period of learning. However, the whittle index policy relies heavily on the prior system information, while our scheme is model-free. Due to the stochastics of channel states, there may not be any available channel in one time slot. Thus, we count the idle channels in all time slots and get the optimal throughputs in case III and case IV are 0.9 and 0.93, respectively.

4.3. Complex Heterogeneous Network Scenarios. For further analysis, we study the performance of our scheme in two more complicated heterogeneous network scenarios.

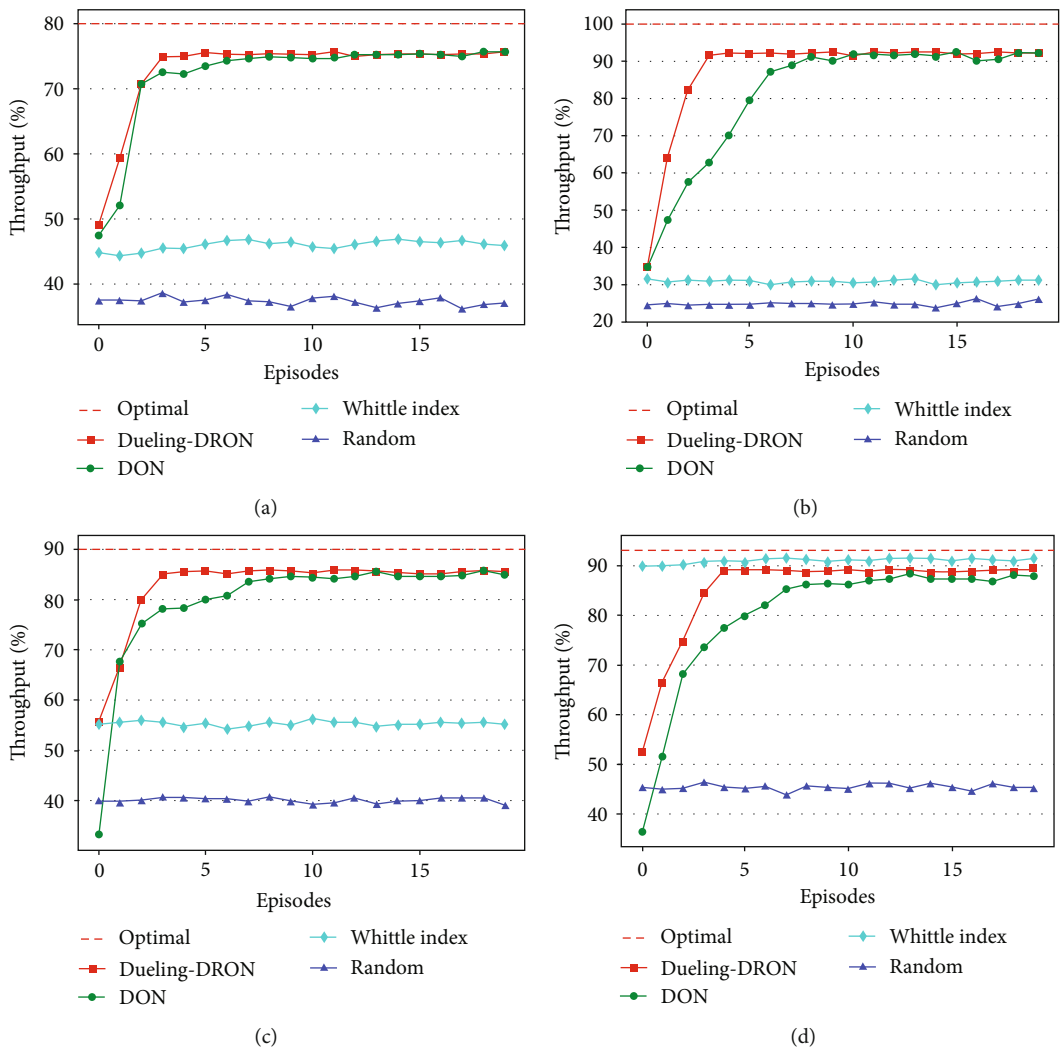


FIGURE 5: The throughput of the DRL node using different spectrum access policies in different cases. (a) Case I. (b) Case II. (c) Case III. (d) Case IV.

Compared with the simple network scenarios, the complex network scenarios contain multiple heterogeneous terminals from different networks, and these terminals coexist each other with different MAC protocols. The network scale becomes lager, and the spectrum states are more complex. The two scenarios are described as follows.

(1) *Complex Scenario I*. The number of channels is set to 16, i.e., $N = 16$. The heterogeneous wireless networks include two authorized nodes, two TDMA nodes, three hopping nodes, and eight q-ALOHA nodes. Two authorized nodes occupy channel 1 and channel 16, respectively. They transmit packets all the time. Two TDMA nodes occupy channel 6 and channel 15, respectively. One TDMA node transmits 15 time slots within a frame of 16 time slots, and the other node transmits 14 time slots within a frame of 16 time slots. Three hopping nodes dynamically occupy channels 2, 3, 4, and 5 with the patterns: $C2C3C4 \longrightarrow C3C4C5 \longrightarrow C4C5C2 \longrightarrow C5C2C3 \longrightarrow C2C3C4$. Eight q-ALOHA nodes occupy the rest channels 7, 8, 9, 10, 11, 12, 13, and 14; and the transmission probability of each q-ALOHA node in different channels is [0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]

(2) *Complex Scenario II*. The number of channels is set to 16, i.e., $N = 16$. The network environment is the same as the complex scenario I basically. The only difference is the two-Markov nodes replace the q-ALOHA nodes and occupy the channels 7, 8, 9, 10, 11, 12, 13, and 14. The transmission transition probability of these nodes on the n-th channel follows a two-state Markov chain

The experimental results are shown in Figure 6. We can see that both Dueling DRQN and DQN can converge after certain episodes. However, the expanding in the size of observation space and action space increases the difficulty

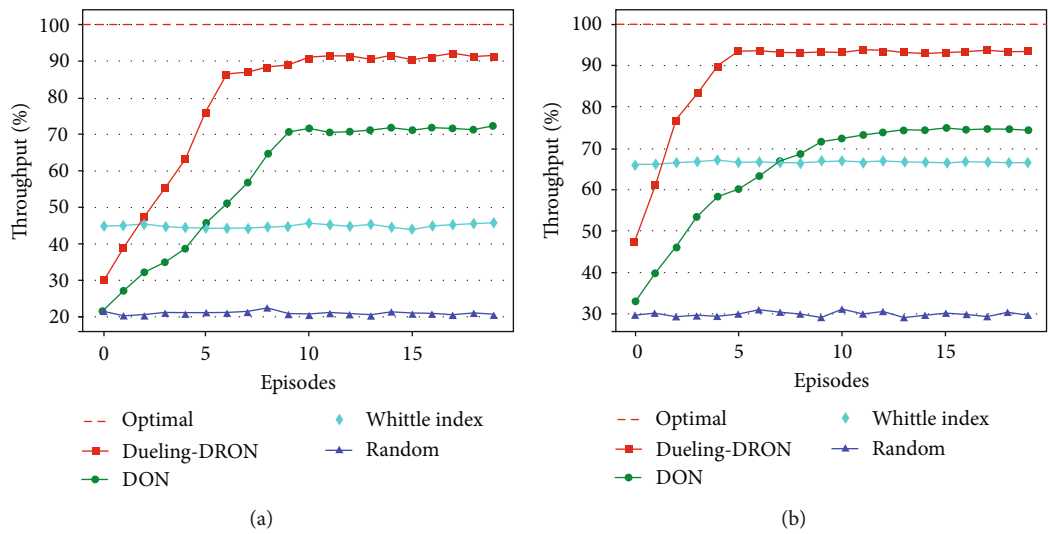


Figure 6: The throughput of the DRL node using different spectrum access policies. (a) Complex scenario I. (b) Complex scenario II.

of learning. The DQN needs to spend more episodes to explore different possible actions and learn the variation characteristics of the spectrum environment, but Dueling DRQN can still learn the spectrum characteristics quickly and obtain a better throughput than DQN. In addition, by comparing the performance of Dueling DRQN and DQN in both simple scenarios and complex scenarios, we can find that DQN obtains a better performance in the simple scenario but fails in the complex scenario. Unlike the DQN, the Dueling DRQN can keep robust in different scenarios due to the ability to infer and predict the complex temporal correlation of spectrum states. Moreover, we count the channel access frequency of different access strategies during the whole simulation process in two scenarios, and the results are shown in Figure 7. Since the relevance of the channel state is hidden deeply and changes over time, the strategy of DQN is ambiguous. It is very difficult for the DRL node to make an effective decision, especially in a partially observed environment. However, for the Dueling DRQN, by capturing the related information from historical observations and inferring the relevance of channel states, it focuses to access some channels (channel 7 and 8) as many as possible because these two channels have more idle time slots to share with the DRL node than other channels. Thus, the strategy of Dueling DRQN is explicit. In other words, the DQN looks more aggressive but the Dueling DRQN looks more comprehensive. The access strategy range of the whittle index policy is mainly between channel 7 and channel 14 both in two scenarios. However, it performs worse in the scenario I than in the scenario II because these channels in the scenario I do not follow the two-state Markov transition models.

4.4. Time-Varying Network Scenarios. In previous evaluations, the network parameters are fixed in a particular scenario. In this section, we further study the adaptability of the Dueling DRQN policy in a time-varying environment, in which the number of channels and the transmission schedule of nodes will be changed. As shown in Figure 8, we first consider that the number of channels is six, i.e., $N = 6$. The network includes two authorized nodes and four TDMA nodes. Two authorized nodes occupy channel 1 and channel 2, respectively. They transmit packets all the time. Four TDMA nodes occupy the rest channels and transmit $X = 10, 12, 13, 14$ time slots within a frame of $Y = 16$ time slots. In the second state, the transmission schedule of these TDMA nodes will change. Specifically, the order of time slots used by the TDMA nodes in a frame is changed. The detailed changes for the transmission schedule of TDMA nodes are displayed in Table 3. In the third state, the number of channels is increased to eight. Two TDMA nodes depart the network (originally occupy channel 5 and channel 6) and three hopping nodes join the network. Three hopping nodes dynamically occupy channels 5, 6, 7, and 8 with the patterns: $C5C6C7 \longrightarrow C6C7C8 \longrightarrow C7C8C5 \longrightarrow C8C5C6 \longrightarrow C5C6C7$. In the fourth state, the number of channels is increased to 12. Four q-ALOHA nodes join the network, which occupy the new channels in the network, and the transmission probability of each q-ALOHA node on the new channels is [0.4, 0.5, 0.8, 0.9]. In the fifth state, the number of channels is increased to 16. Four two-Markov nodes join the network and occupy the new channels in the network. The transmission transition probability of these nodes on the new channel follows a two-Markov chain. It is noted that the performance metric is reset when the environment changes. We can see that the Dueling DRQN can autonomously adjust to these changes in the environment and relearn the optimal access policy during $4 \sim 6$ episodes. It also demonstrates that the proposed scheme can follow the change in the transmission schedule of nodes without the prior knowledge of the MACs of other nodes. In addition, the optimal throughputs that the DRL node can achieve in different subenvironments are different, and the gap between the proposed algorithm and the optimal access scheme is getting larger as the network environment changes. The reason is that the increasing in the number of nodes and channels will lead to an exponential increase in the space of state. The expansion of state space

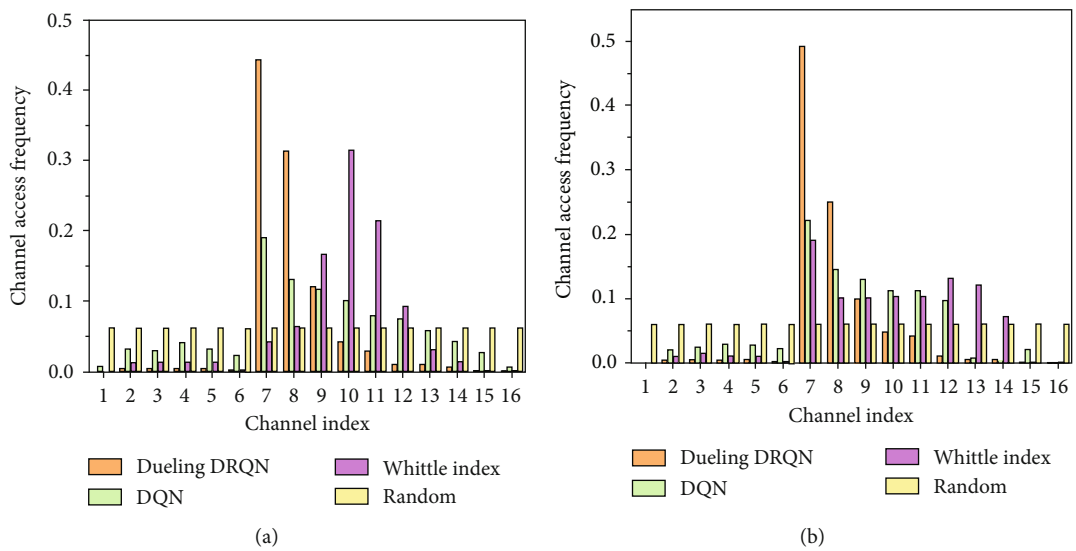


Figure 7: The statistics for channel access frequency of DRL node in different complex heterogeneous scenarios. (a) Complex scenario I. (b) Complex scenario II.

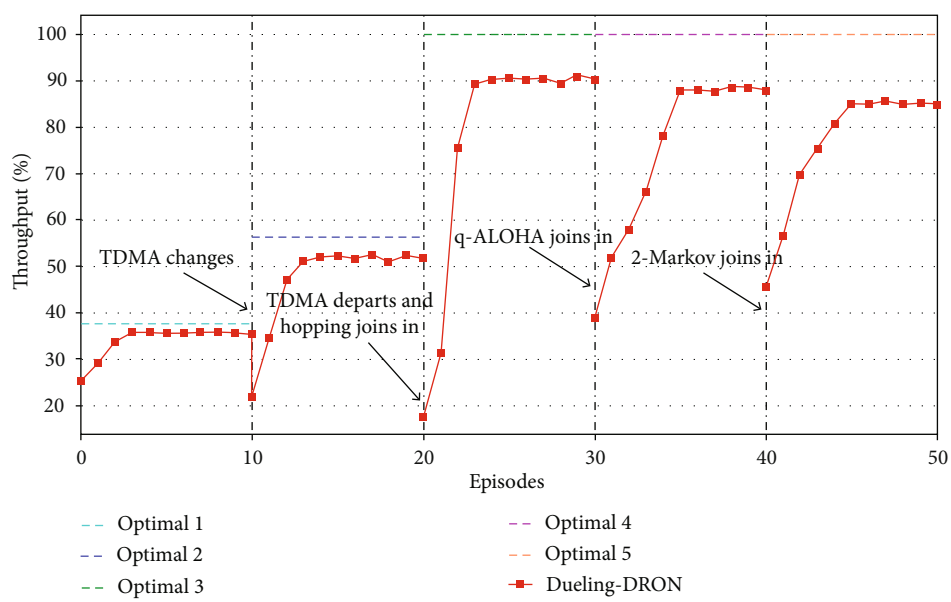


Figure 8: The throughput of the DRL node using Dueling DRQN access policy in the time-varying environment.

Table 3: Changes in transmission schedule of TDMA nodes.

TDMA node 1	Use 1, 2, 3, 7, 8, 9, 10, 13, 14, 15 slots in a frame
TDMA node 2	Use 1, 2, 3, 4, 6, 7, 8, 9, 12, 13, 14, 15 slots in a frame
TDMA node 3	Use 1, 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14, 15 slots in a frame
TDMA node 4	Use 1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15 slots in a frame

will greatly increase the complexity of learning the optimal access strategies. In addition, the temporal feature of spectrum states will be more difficult to learn and capture by the agent as the network environment changes. Last but not least, the collision between the DRL node and other heterogeneous nodes may surge with the increase of the number of nodes and channels, which means the transmission failure of the DRL node. Statistically, the optimal throughputs for different subenvironments are 0.375, 0.5625, 1, 1, and 1, respectively.

4.5. Computational Complexity Analysis. In the last section, we give the specific computational complexity analysis of the proposed Dueling DRQN algorithm. The complexity of LSTM is closely related with the "cell" which is the basic computing component of LSTM. According to [36], the complexity of an LSTM at each time step can be written as $C = 4 \times n_c^2 + 4 \times n_i \times n_c + n_c \times n_{i+1} + 3 \times n_c$, where n_c is the number of memory cells, i.e., the number of memory blocks. n_i is the number of neural units in the last layer, and n_{i+1} is the number of neural units in the following layer. The rest of the neural networks in Figure 4 is fully connected, and the computational complexity of these neural networks can be represented by the number of multiplications in them. The number of multiplications through the Dueling DRQN with M layers (except LSTM layer) is $D = U \cdot m_1 + \sum_{j=1}^{M-2} m_j \cdot m_{j+1} + L \cdot m_{M-1}$, where U is the size of the input layer, m_j is the number of neural units in the j-th layer, and L is the size of the output layer. Thus, the complexity of our proposed Dueling DRQN is given by $O(C + D)$ at each time step.

5. Conclusion

In this paper, we investigated the problem of the dynamic multichannel access in heterogeneous wireless networks and proposed a DRL framework based on the combination of DQN, LSTM, and Dueling architecture, namely, Dueling DRQN. In the Dueling DRQN framework, the DQN is used to explore the idle spectrum resources, the LSTM is employed to make full use of historical observations and infer the temporal features of the heterogeneous networks, and the Dueling architecture is introduced to overcome the observability problem about dynamic environment in neural networks. We examined our scheme in various heterogeneous scenarios and compared it with other baseline channel access schemes such as the DQN access policy, the whittle index policy, and the random access policy. Simulation results showed that the Dueling DRQN can achieve a better throughput improvement than these baseline schemes and keep robust in complex heterogeneous network scenarios. Besides, through simulating our Dueling DRQN in the time-varying scenario, we found that our proposed scheme can also adapt well to the time-varying environment without any artificial adjustment or prior knowledge about the environment. For the future work, we will investigate the model-free dynamic multichannel access methods for multiple intelligent nodes without extra message interaction in the heterogeneous wireless networks.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant nos. 61931020, 62001483, and 62171449.

References

[1] H. Yang, "Application and development of mobile communication technology," in *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 893–896, Harbin City, China, 2021.

[2] Z. Ning, S. Sun, X. Wang et al., "Intelligent resource allocation in mobile blockchain for privacy and security transactions: a deep reinforcement learning based approach," *Science China (Information Sciences)*, vol. 64, no. 6, pp. 172–187, 2021.

[3] Y. Liu, Q. Sun, Y. Tang, Y. Li, W. Jiang, and J. Wu, "Virtual reality system for industrial training," in *2020 International Conference on Virtual Reality and Visualization (ICVRV)*, pp. 338-339, Vigo, Spain, 2020.

[4] R. H. Tehrani, S. Vahid, D. Triantafyllopoulou, H. Lee, and K. Moessner, "Licensed spectrum sharing schemes for mobile operators: a survey and outlook," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2591–2623, 2016.

[5] Z. Ning, Y. Yang, X. Wang et al., "Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[6] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[7] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, pp. 1–9, New York, NY, USA, 2016.

[8] K. Wang, L. Chen, Q. Liu, and K. Al Agha, "On optimality of myopic sensing policy with imperfect sensing in multi-channel opportunistic access," *IEEE Transactions on Communications*, vol. 61, no. 9, pp. 3854–3862, 2013.

[9] K. Liu and Q. Zhao, "Indexability of restless bandit problems and optimality of whittle index for dynamic multichannel access," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5547–5567, 2010.

[10] H. Zhou, B. Liu, F. Hou et al., "Database-assisted dynamic spectrum access with QoS guarantees: a double-phase auction approach," *China Communications*, vol. 12, no. 1, pp. 66–77, 2015.

[11] F. Li, K. -Y. Lam, L. Meng, H. Luo, and L. Wang, "Trading-based dynamic spectrum access and allocation in cognitive internet of things," *IEEE Access*, vol. 7, pp. 125952–125959, 2019.

[12] S. Barrachina-Muñoz, A. Chiumento, and B. Bellalta, "Multi-armed bandits for spectrum allocation in multi-agent channel bonding WLANs," *IEEE Access*, vol. 9, pp. 133472–133490, 2021.

[13] Y. Zhang, Q. Zhang, B. Cao, and P. Chen, "Model free dynamic sensing order selection for imperfect sensing multichannel cognitive radio networks: a Q-learning approach," in *2014 IEEE International Conference on Communication Systems*, pp. 364–368, Macau, China, 2014.

[14] C. Dhahri and T. Ohtsuki, "Q-learning cell selection for femtocell networks: single- and multi-user case," in *2012 IEEE Global Communications Conference (GLOBECOM)*, pp. 4975–4980, Anaheim, CA, USA, 2012.

[15] K. Malon, J. Łopatka, and P. Skokowski, "Q-learning based radio channels utility evaluation algorithm for the local dynamic spectrum management in mobile ad-hoc networks," in *2020 Baltic URSI Symposium (URSI)*, pp. 28–32, Warsaw, Poland, 2020.

[16] N. Morozs, D. Grace, and T. Clarke, "Distributed Q-learning based dynamic spectrum access in high capacity density cognitive cellular systems using secondary LTE spectrum sharing," in *2014 International Symposium on Wireless Personal Multimedia Communications (WPMC)*, pp. 462–467, Sydney, NSW, Australia, 2014.

[17] Z. Ning, S. Sun, X. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[18] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[19] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 2, pp. 257–265, 2018.

[20] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "A deep actor-critic reinforcement learning framework for dynamic multichannel access," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1125–1139, 2019.

[21] S. B. Janiar and V. Pourahmadi, "Deep-reinforcement learning for fair distributed dynamic spectrum access in wireless networks," in *2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–4, Las Vegas, NV, USA, 2021.

[22] P. Yang, L. Li, J. Yin et al., "Dynamic spectrum access in cognitive radio networks using deep reinforcement learning and evolutionary game," in *2018 IEEE/CIC International Conference on Communications in China (ICCC)*, pp. 405–409, Beijing, China, 2018.

[23] M. J. Liston and K. R. Dandekar, "Entropy based exploration in cognitive radio networks using deep reinforcement learning for dynamic spectrum access," in *2021 IEEE 21st Annual Wireless and Microwave Technology Conference (WAMICON)*, pp. 1–5, Sand Key, FL, USA, 2021.

[24] P. Chen, S. Guo, and Y. Gao, "Deep reinforcement learning with bidirectional recurrent neural networks for dynamic spectrum access," in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, pp. 1–5, Norman, OK, USA, 2021.

[25] E. F. Badran, A. A. Bashir, A. I. Zaki, and W. K. Badawi, "Orthogonal codes-based dynamic spectrum access in cognitive radio networks," *China Communications*, vol. 16, no. 12, pp. 34–46, 2019.

[26] Y. Yu, T. Wang, and S. C. Liew, "Deep-reinforcement learning multiple access for heterogeneous wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1277–1290, 2019.

[27] Y. Yu, S. C. Liew, and T. Wang, "Non-uniform time-step deep Q-network for carrier-sense multiple access in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing*, vol. 20, no. 9, pp. 2848–2861, 2021.

[28] H. Yang, J. Zhao, K. -Y. Lam, Z. Xiong, Q. Wu, and L. Xiao, "Distributed deep reinforcement learning based spectrum and power allocation for heterogeneous networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 9, pp. 6935–6948, 2022.

[29] N. Zhao, Y. -C. Liang, D. Niyato, Y. Pei, M. Wu, and Y. Jiang, "Deep reinforcement learning for user association and resource allocation in heterogeneous cellular networks," *IEEE Transactions on Wireless Communications*, vol. 18, no. 11, pp. 5141–5152, 2019.

[30] X. Wang, Z. Ning, S. Guo, M. Wen, and V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3225–3238, 2022.

[31] V. Mnih, K. Kavukcuoglu, D. Silver et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[32] J. Xiao, J. Li, and X. Wang, "Network reconfiguration of the shipboard power system based on logistic function particle swarm optimization," in *2008 7th World Congress on Intelligent Control and Automation*, pp. 5366–5370, Chongqing, 2008.

[33] R. A. Callejas-Molina, V. M. Jimenez-Fernandez, and H. Vazquez-Leal, "Digital architecture to implement a piecewise-linear approximation for the hyperbolic tangent function," in *2015 International Conference on Computing Systems and Telematics (ICCSAT)*, pp. 1–4, Xalapa, Mexico, 2015.

[34] K. Tachibana and K. Otsuka, "Wind prediction performance of complex neural network with ReLU activation function," in *2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pp. 1029–1034, Nara, Japan, 2018.

[35] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[36] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *15th Annual Conference of the International Speech Communication Association*, pp. 1–5, 2014.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7410708, 11 pages
https://doi.org/10.1155/2022/7410708

Research Article

A 3D REM-Guided UAV Path Planning Method under Communication Connectivity Constraints

Xingguang Liu, Li Zhou, Xiaoying Zhang, Xiang Tan, and Jibo Wei

College of Electronic Science and Technology, National University of Defense Technology, Changsha, Hunan 410073, China

Correspondence should be addressed to Li Zhou; zhouli2035@nudt.edu.cn

Received 22 July 2022; Accepted 14 September 2022; Published 30 September 2022

Academic Editor: Amr Tolba

With the emergence of a large number of smart devices, the radio environment in which unmanned aerial vehicles (UAVs) take tasks is becoming more and more complex, which puts forward higher requirements for UAVs' situational awareness and autonomous obstacle avoidance capabilities. To tackle this issue, we propose a three-dimension (3D) UAV path planning method under communication connectivity constraints guided by radio environment maps (REMs), which are distributed by ground edge servers in the form of compressed global REMs and detailed local REMs. An interfered fluid dynamic system (IFDS) model is deployed on UAVs to allow them to avoid obstacles and plan paths. We propose a twin-delayed deep deterministic policy gradient- (TD3-) based deep reinforcement learning (DRL) method to optimize the reaction coefficients of UAVs to avoid obstacles and improve the signal to interference plus noise ratio (SINR). The simulation results show that the proposed algorithm can effectively avoid static obstacles and dynamic interference under communication connectivity constraints, significantly improve the communication stability with a higher receive signal SINR and reduce the cost of UAV performing tasks with the shortest path.

1. Introduction

Due to the rapid innovation and technological subversion of the unmanned aerial vehicle (UAV) manufacturing industry, more and more UAVs are being used for aerial surveillance, air cargo, and interference monitoring. And UAVs must not only avoid space obstacles but also avoid radio interference so that communication functions can be maintained [1]. The radio environment faced by UAVs has the characteristics of obstacle (interference) intensive, dynamic, and uncertain. The complex radio environment brings great challenges to the flight safety of UAVs and also puts forward higher requirements for the autonomous control capabilities of UAVs. It has become a research issue for UAVs to recognize the complex radio environment and improve their autonomous obstacle avoidance ability.

Radio environment map (REM) is an important tool for awareness of complex radio environments. It combines geographic terrain coordinates, communication policies, radio environment parameters, and other related information to describe the radio environment from multiple dimensions such as time, frequency, space, and power [2]. REM can assist UAVs in cognition of space and radio environment, plan paths efficiently, or avoid obstacles in real-time. [3] proposed a 3D REM-assisted UAV path design method. By combining the spatial 3D map and the radio propagation model, the UAV is assisted in designing a path that maintains a cellular connection. However, the coverage area of the base stations is defined as the same altitude as the UAV, which is actually a 2D path planning problem. In [4], the UAV constructs REM through the synergies between vision and communication in the edge network, which assists the UAV in the realization of online path planning and autonomous flight. The authors in [5] proposed a UAV path planning method, which exploits a compressed global map of the environment combined with a cropped but uncompressed local map showing the vicinity of the UAV. This method of distributing global and local map information for UAV path planning is inspired. In path planning, the distant information will cause general direction decisions, and close information will cause immediate action, such as avoiding obstacles. Therefore, the details of

the distant obstacles can be less than the details of the surrounding obstacles for UAVs. The above work has not carried out detailed research on the path planning and autonomous obstacle avoidance method of UAVs.

The traditional path planning methods mainly include the model predictive control [6], optimization algorithm [7–9], stochastic programming [10], and geometric calculation [11]. However, these methods are designed to deal with 2D path planning. When extended to 3D path planning, the amount of computation will increase exponentially, and the paths generated in discrete environments have poor smoothness. An additional smoothing algorithm is required to optimize the path, which increases the complexity of the path planning algorithm. The authors in [12] proposed the artificial potential field (APF) method based on the concept of force field in physics. The goal point has a "gravitational force" on the UAV, and the obstacles have "repulsive forces" on the UAV. Finally, by calculating the resultant force to control the movement of the UAV, this method is suitable for 3D path planning, which has low computational complexity and can plan a smooth path. However, the method could also fall into a local optimum at certain locations and even get into the interior of obstacles. In [13], the interfered fluid dynamical system (IFDS) model was proposed for the first time, which draws on the macroscopic characteristics of natural water flow. When there are no obstacles, the water flows in a straight line. While encountering an obstacle, the water would smoothly bypass the obstacle. The algorithm has low computational complexity and can handle complex radio interference and obstacles of different shapes.

In the complex spatial obstacle and radio interference environment (such as the coexistence of static and dynamic obstacles of different shapes and sizes), the position of the obstacle changes dynamically, and the environmental information must be updated in real-time. And it is also necessary to optimize the reaction coefficients of the IFDS model to get the best path for the UAV which is surrounded by obstacles so that the UAV flight path is the shortest. In [14], the neural network is used to optimize the reaction coefficients of the IFDS model. The relative positions between UAV, the destination and obstacles are extracted from the sample data as the input of the neural network, and the reaction coefficient of the IFDS model is used as the output of the neural network. The authors in [15, 16] adopt the deep reinforcement learning (DRL) algorithm to optimize the reaction coefficients, which retains the advantages of the analytical method and maintains a high calculation speed. The algorithm has great application potential.

(1) In this paper, we propose a 3D REM-guided path planning method for UAVs in order to improve the environmental awareness and autonomous obstacle avoidance capabilities of UAVs. The ground edge server distributes the compressed global REM to the UAV before the UAV launches. The UAV adapts the IFDS model to preplan a path according to the global REM and starts to fly. When the edge server detects an obstacle coming within a safe distance of the UAV, it distributes a cropped but uncompressed detailed local map for the UAV. Then the UAV adapts the IFDS model to avoid obstacles efficiently and optimize the reaction coefficients of IFDS based on the twin-delayed deep deterministic policy gradient (TD3) algorithm to obtain the shortest path and improve the signal to interference plus noise ratio (SINR). Our principal contributions are summarized as follows

(1) We propose a 3D REM-guided path planning method for UAVs. The compressed global REM provides the UAV with global spatial obstacle and radio interference information and preplans a path for the UAV. When an obstacle or interference is detected, the cropped but uncompressed detailed local map is distributed to the UAV to avoid interference. The method makes the SINR of the UAV received signal exceed the interference threshold to avoid losing communication with the base station

(2) We propose an obstacle avoidance model for UAVs based on IFDS. By adjusting the repulsive reaction coefficient, the tangential reaction coefficient and the tangential direction coefficient, the path of the UAV in the 3D space is optimized

(3) To optimize the reaction coefficients in the IFDS model, we propose a DRL algorithm based on TD3, which makes the UAV flight path the shortest while satisfying the communication connectivity constraints

The remaining sections of this paper are organized as follows. Section 2 describes the global and local REM construction and distribution methods. Then, the IFDS model and problem formulation are introduced in Section 3. Section 4 specifies the implementation details of our TD3-based DRL algorithm. Performance evaluations are provided in Section 5, and Section 6 concludes the paper.

2. The Construction and Distribution Method of REMs

We consider a UAV flying from the starting point to the goal point in an edge network, as shown in Figure 1. The UAV needs to maintain communication with the base station with an edge server while avoiding obstacles and interference to reach the goal point in the complex radio environment.

We proposed a global and local REM construction and distribution method for UAV obstacle avoidance and path planning. In the complex radio environment, devices with sensing function such as UAVs, fixed monitoring stations, vehicle-mounted receivers, handheld spectrum analyzers, and other devices are deployed to sense spectrum data in 3D space and upload the data to the edge server [17]. According to the collected data, the edge server adopts the Kriging interpolation algorithm [18] to interpolate the unknown point. The spectrum data of the unknown points can be calculated as

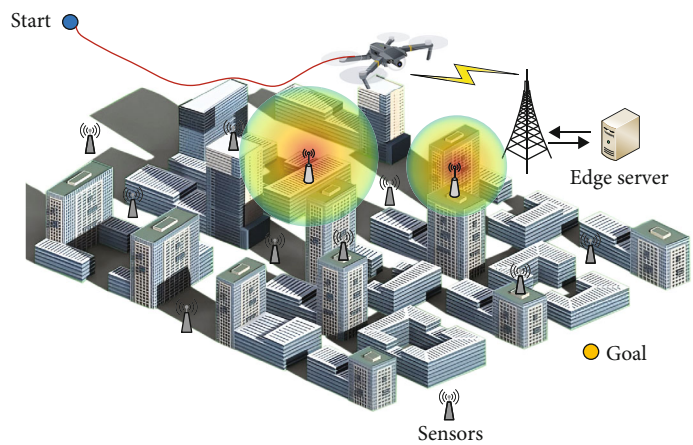


Figure 1: The networking scene of the UAV flight.

$$\hat{s}_0 = \sum_{i=1}^{n} \omega_i s_i, \tag{1}$$

where $\{s_1, s_2, \cdots, s_n\}$ is the spectral data sensed by the sensor devices and ω_i is the weight of the data sensed by the sensors to the unknown point $\hat{s}_0$.

In the complex radio environment, spectrum resources and available bandwidth are limited. If the edge server directly distributes a high-resolution global REM, it will cause a great burden on the communication bandwidth and high delay, which may cause the UAV to lose communication or collide with obstacles due to sudden radio interference. To tackle the issue, we propose a method of sending a low-resolution global REM and a high-resolution local REM for UAV path planning.

The edge server performs low-resolution interpolation based on the collected spectrum data before the UAV launches, which analyzes the location of radio interference and space obstacles. Then the edge server distributes a compressed global REM to the UAV. When a sudden interference or dynamic obstacle occurs, the edge server distributes a high-resolution local REM to the UAV periodically, so that the UAV can avoid the obstacle in real-time according to the position and threat level of the interference or obstacle. The 3D REM-guided path planning method implemented in the UAV is shown in Figure 2.

The edge server analyzes and extracts features from the 3D REM and identifies space obstacles. Then, we set an SINR threshold based on the UAV's received signal from the base station and interfering signals. And the edge server abstracted radio interference as spheres and spatial obstacles as spheres, cones, and cylinders. Spatial obstacles and radio interferences in the environment can be equivalent to the standard convex envelope equation

$$\Gamma(\mathbf{P}) = \left(\frac{x - x_k}{a + r_A}\right)^{2m} + \left(\frac{y - y_k}{b + r_A}\right)^{2n} + \left(\frac{z - z_k}{c + r_A}\right)^{2l}, \tag{2}$$

where $a, b, c > 0$ determine the size of the obstacle, $m, n, l > 0$ control the shape of the obstacle. When $m = n = l = 1$ and $a = b = c$, the obstacle is a sphere. When $m = n = 1$, $l > 1$, and $a = b$, the obstacle is a cylinder. When $m = n = 1$, $0 < l < 1$, and $a = b$, the obstacle is approximately a cone. (x_k, y_k, z_k) represents the center coordinate $\mathbf{P}_k$ of the obstacle k. r_A denotes the safe distance of the UAV. $\Gamma(\mathbf{P}) > 1$, $\Gamma(\mathbf{P}) = 1$, and $\Gamma(\mathbf{P}) < 1$ express that the UAV position $\mathbf{P}$ is located outside on the surface and inside the equivalent envelope of the obstacle, respectively.

3. IFDS Model and Problem Formulation

3.1. IFDS Model. A UAV with velocity V flies from the current position to the goal point $\mathbf{P}_g = (x_g, y_g, z_g)$, and the Euclidean distance between the two points can be calculated as $\mathrm{dist}(\mathbf{P}, \mathbf{P}_g) = \sqrt{(x - x_g)^2 + (y - y_g)^2 + (z - z_g)^2}$. When there are no obstacles on the path of the UAV flying from $\mathbf{P}$ to the goal point $\mathbf{P}_g$, the initial flow field is a straight line, and the initial flow velocity of the UAV can be denoted as

$$\mathbf{u}(\mathbf{P}) = \left[\frac{V(x_d - x)}{\mathrm{dist}(\mathbf{P}, \mathbf{P}_g)} \; \frac{V(y_d - y)}{\mathrm{dist}(\mathbf{P}, \mathbf{P}_g)} \; \frac{V(z_d - z)}{\mathrm{dist}(\mathbf{P}, \mathbf{P}_g)}\right]^T. \tag{3}$$

When there are K obstacles in the environment, the weighted sum of the interference matrix of all obstacles to the UAV is indicated as

$$\bar{\mathbf{M}}(\mathbf{P}) = \sum_{k=1}^{K} \omega_k(\mathbf{P}) \mathbf{M}_k(\mathbf{P}), \tag{4}$$

where $\omega_k(\mathbf{P})$ represents the weight of k-th obstacle. It is determined by the distance from the UAV to the equivalent envelope of the obstacle. The larger the distance, the smaller the weight, and the smaller the interference effect on the UAV

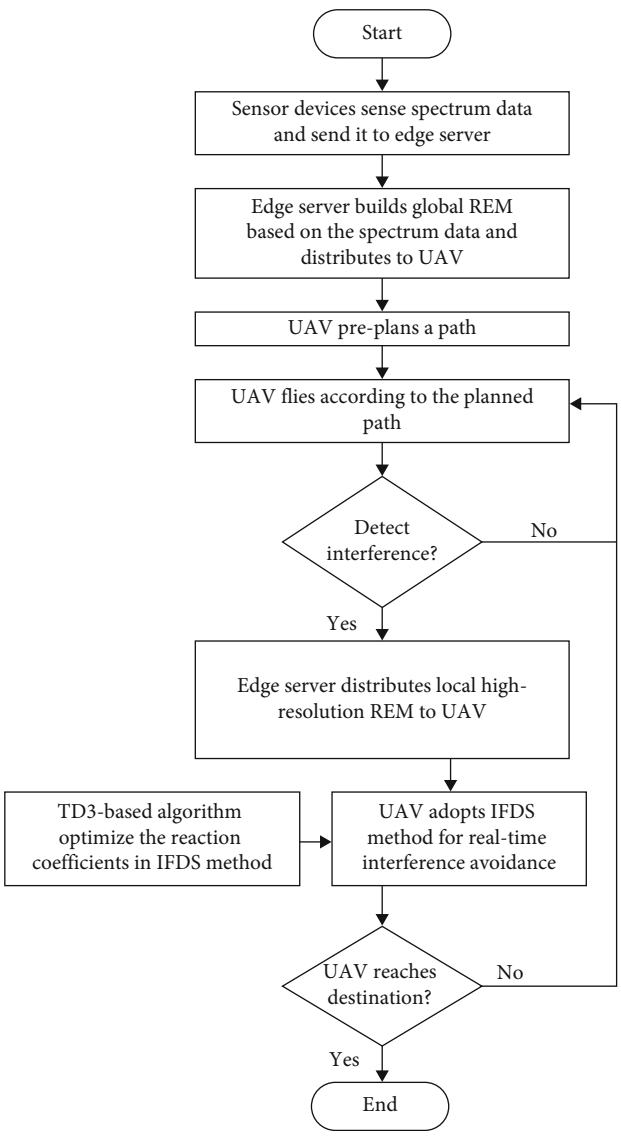


Figure 2: The 3D REM-guided path planning method.

$$\omega_k(\mathbf{P}) = \begin{cases} 1, & K = 1, \\ \prod_{i=1, i\neq k}^{K} \dfrac{(\Gamma_i(\mathbf{P}) - 1)}{(\Gamma_i(\mathbf{P}) - 1) + (\Gamma_k(\mathbf{P}) - 1)}, & K \neq 1. \end{cases} \tag{5}$$

The interference matrix of obstacle k can be calculated as

$$\mathbf{M}_k(\mathbf{P}) = \mathbf{I} - \frac{\mathbf{n}_k(\mathbf{P})\mathbf{n}_k^T(\mathbf{P})}{|\Gamma_k(\mathbf{P})|^{1/\rho_k}\mathbf{n}_k^T(\mathbf{P})\mathbf{n}_k(\mathbf{P})} + \frac{\mathbf{t}_k(\mathbf{P})\mathbf{n}_k^T(\mathbf{P})}{|\Gamma_k(\mathbf{P})|^{1/\sigma_k}\mathbf{t}_k(\mathbf{P})\mathbf{n}_k(\mathbf{P})}, \tag{6}$$

where $\mathbf{I}$ is the unit attraction matrix. The second and third terms of Equation (6) are the repulsion matrix and the tangential matrix, respectively. ρ_k and σ_k correspondingly denote the repulsive reaction coefficient and tangential reaction coefficient of the UAV to the obstacle k, which determines the timing and safe distance for the UAV to avoid obstacles. $\mathbf{n}_k(\mathbf{P})$ is the vertical vector from the UAV to the obstacle surface, which can be expressed as

$$n_k(\mathbf{P}) = \left[\frac{\delta\Gamma_k(\mathbf{P})}{\delta x}\frac{\delta\Gamma_k(\mathbf{P})}{\delta y}\frac{\delta\Gamma_k(\mathbf{P})}{\delta z}\right]^T. \tag{7}$$

$\mathbf{t}_k(\mathbf{P})$ represents the tangent matrix perpendicular to the vertical vector and tangent to the equivalent envelope surface of the obstacle k, which is derived as

$$\begin{aligned} \mathbf{t}_{k,1}(\mathbf{P}) &= \left[\frac{\partial\Gamma_k(\mathbf{P})}{\partial y}, -\frac{\partial\Gamma_k(\mathbf{P})}{\partial x}, 0\right]^T, \\ \mathbf{t}_{k,2}(\mathbf{P}) &= \left[\frac{\partial\Gamma_k(\mathbf{P})}{\partial x}\frac{\partial\Gamma_k(\mathbf{P})}{\partial z}, \frac{\partial\Gamma_k(\mathbf{P})}{\partial y}\frac{\partial\Gamma_k(\mathbf{P})}{\partial z}, \right. \\ &\quad \left. -\left(\frac{\partial\Gamma_k(\mathbf{P})}{\partial x}\right)^2 - \left(\frac{\partial\Gamma_k(\mathbf{P})}{\partial y}\right)^2\right]^T. \end{aligned} \tag{8}$$

A coordinate system is established with $\mathbf{t}_{k,1}(\mathbf{P})$, $\mathbf{t}_{k,2}(\mathbf{P})$, and $\mathbf{n}_k(\mathbf{P})$ as the x', y', and z' axes, respectively. Any unit tangent vector in the tangent plane can be denoted as

$$\mathbf{t}'_k(\mathbf{P}) = [\cos\theta_k \ \ \sin\theta_k \ 0]^T, \tag{9}$$

where $\theta_k \in [-\pi, \pi]$ is the angle from the tangent vector to the x'-axis. It determines the direction of the UAV around the obstacle. The tangent vector $\mathbf{t}'_k(\mathbf{P})$ in the O'-$x'y'z'$ coordinate system can be transformed as $\mathbf{t}_k(\mathbf{P})$ of the original coordinate system through the coordinate transformation matrix $\Omega_T^I(\mathbf{P})$, which can be calculated as

$$\mathbf{t}_k(\mathbf{P}) = \mathbf{\Omega}_T^I(\mathbf{P})\mathbf{t}'_k(\mathbf{P}). \tag{10}$$

The coordinate transformation matrix $\mathbf{\Omega}_T^I(\mathbf{P})$ can be represented as

$$\mathbf{\Omega}_T^I(\mathbf{P}) = \begin{bmatrix} \dfrac{r_y}{r_2} & \dfrac{r_x r_z}{r_2 r_3} & \dfrac{r_x}{r_3} \\ \dfrac{r_x}{r_2} & \dfrac{r_y r_z}{r_2 r_3} & \dfrac{r_y}{r_3} \\ 0 & -\dfrac{r_2}{r_3} & \dfrac{r_z}{r_3} \end{bmatrix}, \tag{11}$$

where $r_x = (\partial\Gamma_k(\mathbf{P}))/\partial x$, $r_y = (\partial\Gamma_k(\mathbf{P}))/\partial y$, $r_z = (\partial\Gamma_k(\mathbf{P}))/\partial z$, $r_2 = \sqrt{r_x^2 + r_y^2}$, and $r_3 = \sqrt{r_x^2 + r_y^2 + r_z^2}$. Then the flow velocity can be corrected by the total interference matrix to the initial flow velocity of the UAV

$$\bar{\mathbf{u}}(\mathbf{P}) = \bar{\mathbf{M}}(\mathbf{P})(\mathbf{u}(\mathbf{P}) - v(\mathbf{P})) + v(\mathbf{P}), \tag{12}$$

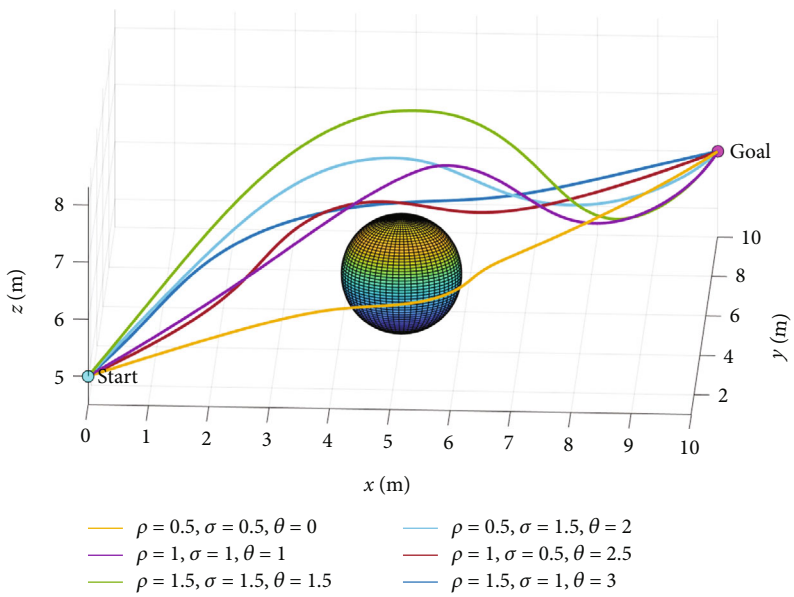


Figure 3: The paths of UAVs by different reaction coefficients in the IFDS model.

where $v(\mathbf{P})$ is the weighted sum of the velocity vectors of all obstacles. It can be indicated as

$$v(\mathbf{P}) = \sum_{k=1}^{K} \omega_k(\mathbf{P}) e^{1-\Gamma_k(\mathbf{P})} v_k, \tag{13}$$

where v_k is the velocity vector of the obstacle k.

3.2. UAV Kinematic Constraints. Since the UAV is affected by the inertia and the delay of REM construction and distribution when moving at high speed, the UAV will move forward for a period of time according to the original velocity before changing the flight state, and this period is the minimum step size $\triangle t$. In addition, due to the limited energy carried by the UAV, the maximum path length that the UAV can fly is $L_{\max}$.

The total time of the UAV from the start point to the goal point is T, the position at time t is $\mathbf{P}_t = \mathbf{P}_{t-1} + \overline{\mathbf{u}_t}\triangle t$, and the flight path length of the UAV at time t can be expressed as

$$L_t = \overline{\mathbf{u}_t}\triangle t, \tag{14}$$

where $\overline{\mathbf{u}_t} = [\overline{\mathbf{u}_x}, \overline{\mathbf{u}_y}, \overline{\mathbf{u}_z}]$ denotes the corrected flow velocity at time t. The climb angle α_t and the yaw angle β_t of the UAV can be calculated as

$$\begin{aligned} \alpha_t &= \arcsin\left(\frac{\overline{\mathbf{u}_z}}{\|\mathbf{u}_t\|}\right), \\ \beta_t &= \arctan\left(\frac{\overline{\mathbf{u}_y}}{\overline{\mathbf{u}_x}}\right). \end{aligned} \tag{15}$$

When the UAV is turning too fast, the yaw angle is too large or the UAV turns sharply, which will cause the UAV to lose its balance and deviate from the original flight path, even cause a crash. And the climb angle of the UAV is related to its own thrust-weight ratio and lift-drag ratio. Therefore, the climb angle and the yaw angle need to satisfy the kinematic constraints

$$\begin{aligned} \alpha'_t &= \alpha_t, \quad |\alpha_t - \alpha_{t-1}| < \alpha_{\max}, \\ \alpha'_t &= \alpha_{t-1} + \alpha_{\max}, \ \alpha_t - \alpha_{t-1} > \alpha_{\max}, \\ \alpha'_t &= \alpha_{t-1} - \alpha_{\max}, \ \alpha_t - \alpha_{t-1} < -\alpha_{\max}, \end{aligned} \tag{16}$$

$$\begin{aligned} \beta'_t &= \beta_t, \quad |\beta_t - \beta_{t-1}| < \beta_{\max}, \\ \beta'_t &= \beta_{t-1} + \beta_{\max}, \ \beta_t - \beta_{t-1} > \beta_{\max}, \\ \beta'_t &= \beta_{t-1} - \beta_{\max}, \ \beta_t - \beta_{t-1} < -\beta_{\max}, \end{aligned} \tag{17}$$

where α'_t and β'_t are the climb angle and the yaw angle after satisfying kinematic constraints. $\alpha_{\max}$ and $\beta_{\max}$ are the maximum constraint angles for the climb and yaw angles, respectively.

3.3. Problem Formulation. The goal of UAV path planning is to adjust the repulsion reaction coefficients, tangential response coefficient, and tangential direction coefficient in the IFDS model to make the UAV flight path the shortest under communication connectivity constraints. Therefore, the objective function of UAV path planning is expressed as

$$\min_{\rho_k, \sigma_k, \theta_k} \int_0^T L_t \, \mathrm{d}t, \tag{18}$$

Initialize: Critic reality network Q_{ξ_1}, Q_{ξ_2} with parameters ξ_1, ξ_2, and actor reality network π_ϕ with parameters ϕ, replay buffer $\mathscr{B}$.
Set target network parameters $\xi'_1 \longleftarrow \xi_1$, $\xi'_2 \longleftarrow \xi_2$, $\phi' \longleftarrow \phi$
for episode = 1 to M **do.**
  Initialize UAV position, get state s_0
  for $t = 1$ **to** T **do:**
    Observe state s_t and select action with exploration noise $a_t \sim \pi_\phi(s_t) + \varepsilon$
    $\varepsilon \sim \mathcal{N}(0, \sigma)$.
    Take action a_t, observe reward r_t and next state s_{t+1}.
    Store (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathscr{B}$.
    Sample a minibatch of Z transitions (s_t, a_t, r_t, s_{t+1}) from $\mathscr{B}$.
    Compute target actions by Equation (24).
    Compute targets by Equation (23).
    Update critic reality network parameters $\xi_j = \underset{\xi_j}{\operatorname{argmin}} Z^{-1} \sum (y - Q_{\xi_j}(s_t, a_t))^2$.
    if t mod d **then:**
    Update ϕ by the deterministic policy gradient.
    $\nabla_\phi J(\phi) = Z^{-1} \sum \nabla_a Q_{\xi_j}(s_t, a_t)|_{a_t = \pi_\phi(s_t)} \nabla_\phi \pi_\phi(s_t)$.
    Update target networks: $\xi'_j \longleftarrow \tau \xi'_j + (1-\tau)\xi'_j$, $\phi' \longleftarrow \tau\phi + (1-\tau)\phi'$.
    end if.
  end for.
end for.

Algorithm 1: TD3-based path planning algorithm.

$$
\begin{aligned}
&C1: \rho_k \in [1,5], \sigma_k \in [1,5], \theta_k \in [-\pi, \pi],\\
&C2: \alpha_t \in [-\pi/2, \pi/2], \beta_t \in [-\pi, \pi],\\
&C3: \begin{cases} \alpha'_t = \alpha_t, & |\alpha_t - \alpha_{t-1}| < \alpha_{\max},\\ \alpha'_t = \alpha_{t-1} + \alpha_{\max}, & \alpha_t - \alpha_{t-1} > \alpha_{\max},\\ \alpha'_t = \alpha_{t-1} - \alpha_{\max}, & \alpha_t - \alpha_{t-1} < -\alpha_{\max}, \end{cases}\\
&C4: \begin{cases} \beta'_t = \beta_t, & |\beta_t - \beta_{t-1}| < \beta_{\max},\\ \beta'_t = \beta_{t-1} + \beta_{\max}, & \beta_t - \beta_{t-1} > \beta_{\max},\\ \beta'_t = \beta_{t-1} - \beta_{\max}, & \beta_t - \beta_{t-1} < -\beta_{\max}, \end{cases}\\
&C5: L = \int_0^T L_t \, \mathrm{d}t < L_{\max},\\
&C6: \mathrm{SINR}_t > \mathrm{SINR}_{\min},
\end{aligned}
\tag{19}
$$

where $C1$ represents the value range of the reaction coefficients, $C2$ indicates the value range of the climb angle and the yaw angle, $C3$ and $C4$ express the kinematic constraints, $C5$ means that the UAV cannot fly more than the longest distance it can fly. $C6$ shows that the receive signal SINR must be higher than the SINR threshold.

As shown in Figure 3, the combination of different coefficients can determine the shape and direction of the path. In previous researches [14–16], receding horizon control (RHC) strategy was mostly used to optimize these coefficients online. However, the serial solution mechanism of RHC cannot well meet the real-time requirements in complex radio environments. Therefore, in this paper, the DRL algorithm is adopted to optimize the coefficients in the IFDS model, so that the path planned by the UAV to avoid obstacles is the shortest.

Table 1: Simulation parameters.

Parameters	Value
The transmit power of the base station	10 W
The transmit power of the interference source	100 mW
The SINR threshold ($\mathrm{SINR}_{\min}$)	-20 dB
The safe distance (r_A)	0.2 km
The time step ($\triangle t$)	0.1 s
The maximum path length that the UAV can fly ($L_{\max}$)	25 km
The maximum climb angle ($\alpha_{\max}$)	10°
The maximum yaw angle ($\beta_{\max}$)	10°
The discount factor (γ)	0.99
The replay buffer size ($\mathscr{B}$)	1×10^6
The batch size (Z)	512

4. TD3-Based Path Planning Algorithm

According to the above objective function, we propose a DRL algorithm based on TD3 to optimize the repulsion reaction coefficient, tangential reaction coefficient, and tangential direction coefficient in the IFDS model. This section first defines the state space, action space, and reward function of the DRL algorithm. Then we introduce the proposed TD3-based path planning algorithm in detail.

4.1. State Space, Action Space, and Reward Function. According to the IFDS model, the state space, action space, and reward function are defined as follows.

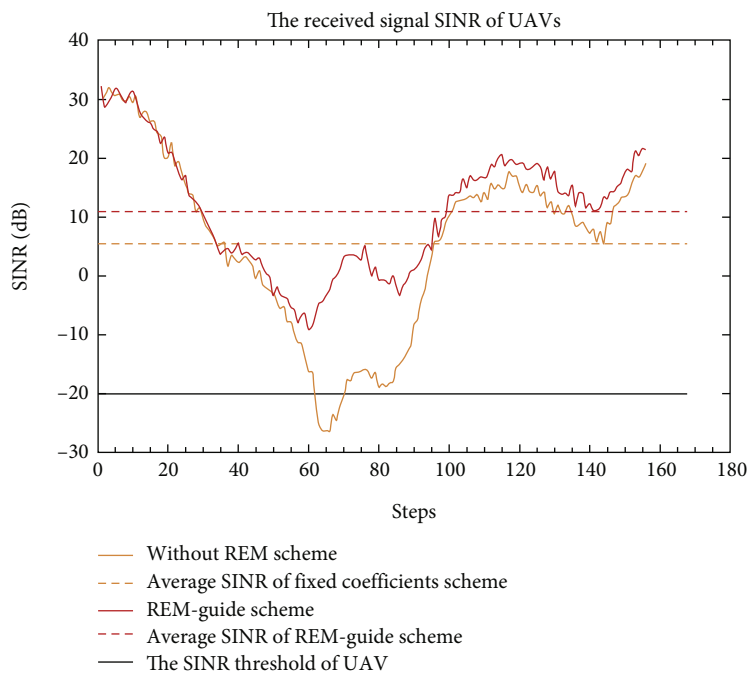


Figure 4: The SINR of REM-guided scheme and Without REM scheme.

(a) State space

The state space can be presented by

$$s_t = \left[\Delta x_{t,k}, \Delta y_{t,k}, \Delta z_{t,k}, \Delta V_{t,xk}, \Delta V_{t,yk}, \Delta V_{t,zk}, \Delta L_{t,k}, \alpha_t, \beta_t\right], \quad (20)$$

where $(\Delta x_{t,k}, \Delta y_{t,k}, \Delta z_{t,k})$ denotes the relative position of the UAV and the obstacle k at time t, $(\Delta V_{t,xk}, \Delta V_{t,yk}, \Delta V_{t,zk})$ expresses the relative velocity of the UAV and the obstacle k, $\Delta L_{t,k}$ indicates the distance from the UAV to the surface of the obstacle k, and α_t and β_t represent the climb angle and the yaw angle of the UAV, respectively.

(b) Action space

The action space can be denoted as

$$a_t = \left[\rho_{t,k}, \sigma_{t,k}, \theta_{t,k}\right], \quad (21)$$

where $\rho_{t,k}, \sigma_{t,k}, \theta_{t,k}$ correspondingly indicate the repulsion reaction coefficient, tangential reaction coefficient, and tangential direction coefficient of the obstacle k at time t. The flying velocity and path of the UAV are affected by adjusting the reaction coefficient in the IFDS model.

(c) Reward function

Generally, the goal of DRL is to maximize the reward, and our goal is to minimize the flight path of the UAV. So the immediate reward is defined as

$$r_t = \frac{\mathrm{Pow}_s}{\mathrm{Pow}_I + p_{\mathrm{noise}}} - \frac{\mathrm{dist}\left(\mathbf{P}_t, \mathbf{P}_g\right)}{\mathrm{dist}\left(\mathbf{P}_0, \mathbf{P}_g\right)}, \quad (22)$$

where Pow_s is the received signal power of the base station, Pow_I denotes the sum of the received signal power of all radio interference, p_{noise} indicates the power of Gaussian noise, $\mathrm{dist}(\mathbf{P}_t, \mathbf{P}_g)$ expresses the distance from the current position to the goal point, and $\mathrm{dist}(\mathbf{P}_0, \mathbf{P}_g)$ represents the distance from the starting point to the goal point.

4.2. TD3-Based Path Planning Algorithm. Considering that there are three continuous variables in the proposed action space, we focus on the policy gradient method, such as the deep deterministic policy gradient (DDPG) algorithm which is often used to deal with continuous action spaces. In [19], the DDPG algorithm is used to optimize the reaction coefficients of the IFDS model.

There are four neural networks in the DDPG algorithm: action reality network, action target network, critic reality network, and critic target network. The parameters of the two critic networks are randomly set, and the parameters of the two action networks are obtained by fitting the input and output. However, the Q function of the critic network in the DDPG algorithm will overestimate the Q values, resulting in the policy invalidation due to the error in the Q function. The TD3 (Twin-Delayed DDPG) algorithm adds two Q functions to each critic network, and the smaller Q value is used as the target in the Bellman error loss function.

$$y = r_t + \gamma \min\left[Q_{\xi_1'}(s_{t+1}, a_t), Q_{\xi_2'}(s_{t+1}, a_t)\right]. \quad (23)$$

In addition, the TD3 algorithm adds noise to the action target network, which makes the policy more difficult to exploit errors in the Q function.

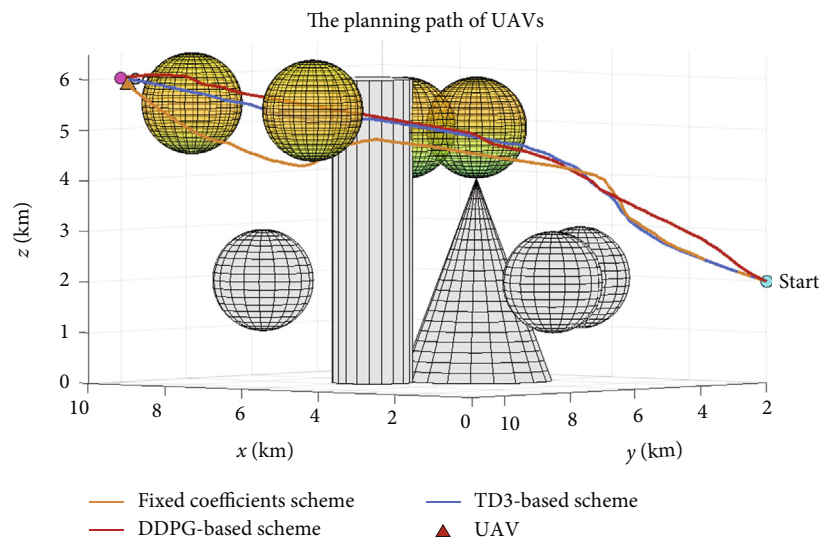
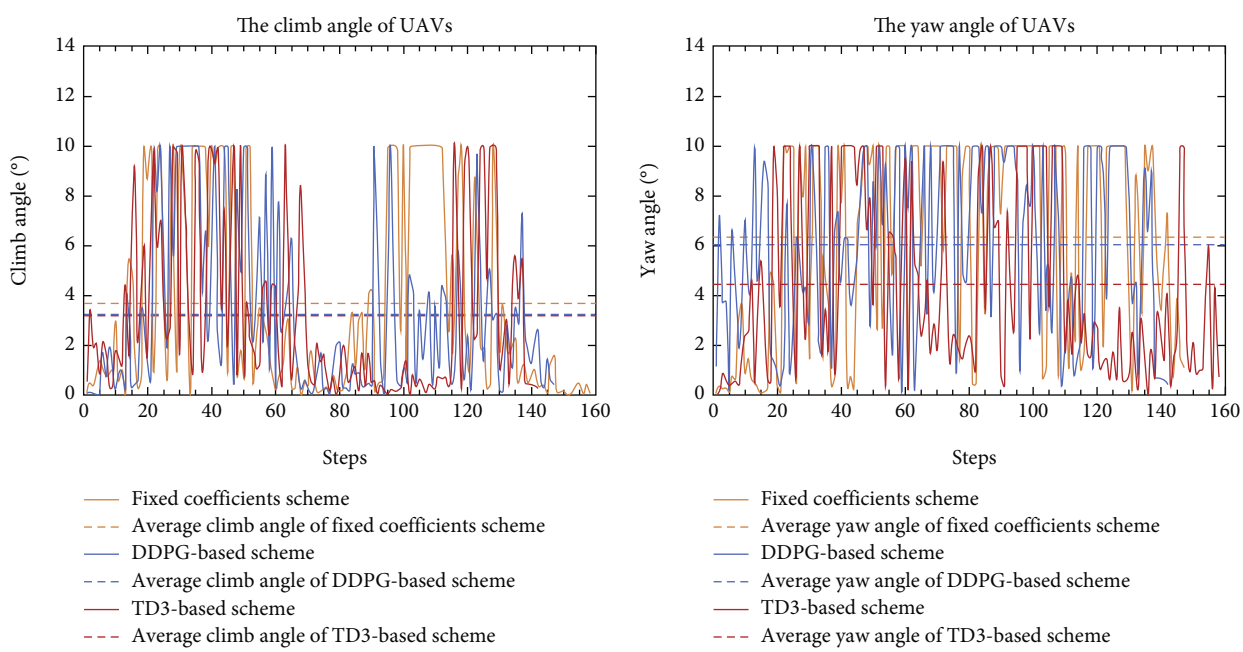


Figure 5: The paths of the three path planning schemes.



(a) The climb angle of UAVs (b) The yaw angle of UAVs

Figure 6: The climb and yaw angles of the three path planning schemes.

$$a'(s_{t+1}) = \text{clip}\left(\pi_\phi(s_{t+1}) + \text{clip}(\epsilon, -c, c), a_{\text{Low}}, a_{\text{High}}\right). \quad (24)$$

Based on the above definitions, the proposed TD3-based path planning algorithm is presented in Algorithm 1. First, we randomly initialize critic reality network parameters ξ_1, ξ_2, and actor reality network parameters ϕ. Then the target network parameters are set as the reality network parameters. Initialize UAV position, get state s_0 in each episode. Observe state s_t and select action with exploration noise $a_t \sim \pi_\phi(s_t) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma)$ at time slot t. Then the UAV take the action a_t to get reward r_t and next state s_{t+1}. The transition (s_t, a_t, r_t, s_{t+1}) is stored in replay buffer $\mathscr{B}$. Finally, sample a minibatch of Z transitions (s_t, a_t, r_t, s_{t+1}) from $\mathscr{B}$ to update the parameters of reality networks and target networks.

5. Simulation Results

5.1. Parameter Setting. In the 3D space of 20 km × 20 km × 20 km, the UAV flies from the start point (0, 2, and 5) to the goal point (10, 10, and 6), and the velocity is 30 m/s, passing multiple obstacles abstracted as sphere, cylinder,

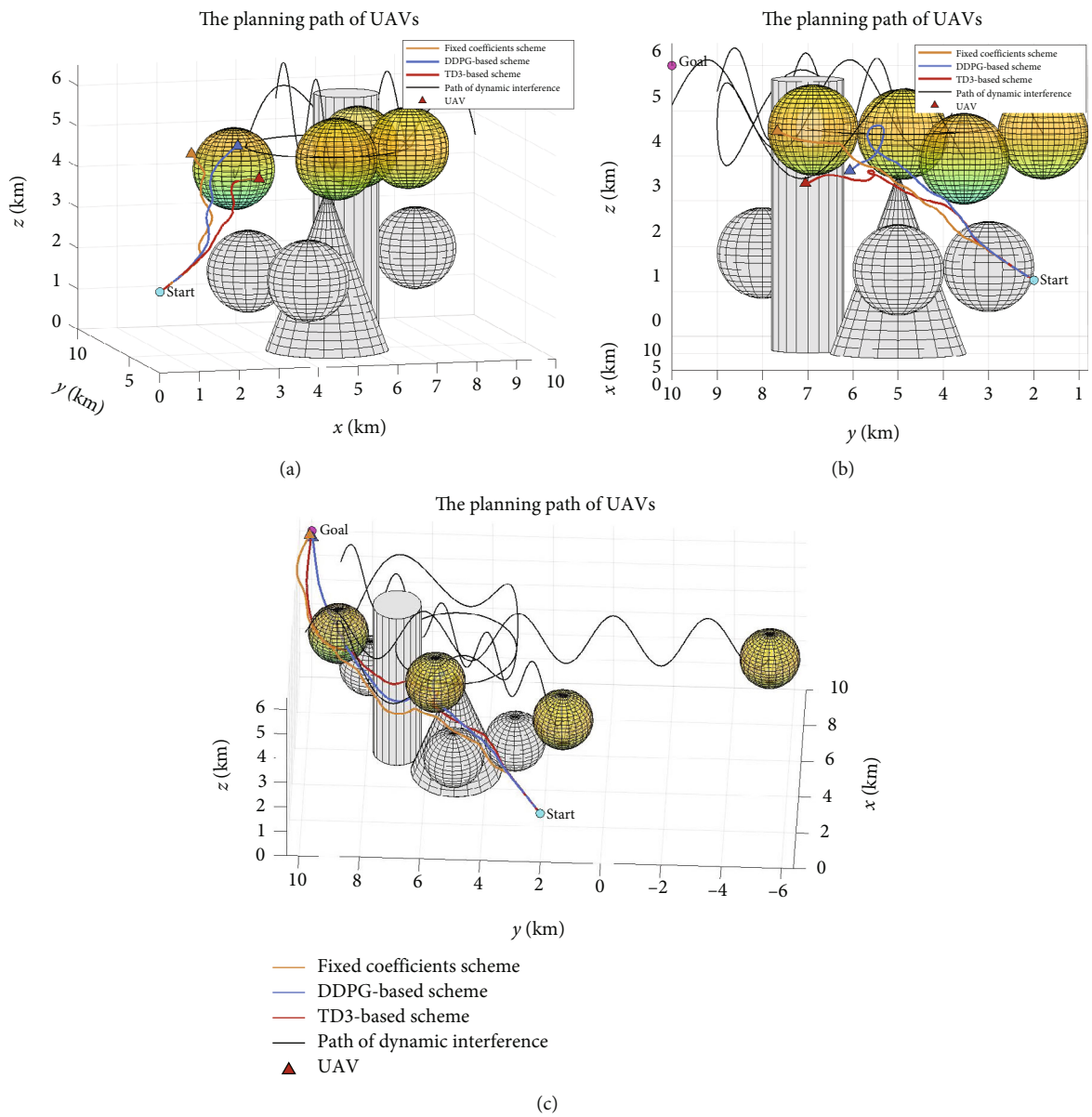


Figure 7: The paths of the three schemes for obstacle avoidance at different time. (a) step = 60, (b) step = 80, and (c) step = 168.

and cone. The base station with an edge server is located at (5, 5, and 0). The transmit power of the base station is 10 W. And the transmit power of the interference source is 100 mW.

We simulate our method in two environments, static and dynamic. The maximum climb angle is $\alpha_{\max} = 10^{\circ}$, the maximum yaw angle is $\beta_{\max} = 10^{\circ}$, and the minimum time step is $\triangle t = 0.1$s. The TD3 algorithm has a discount factor of 0.99, buffer size of $\mathscr{B} = 1 \times 10^6$, and sampling size of $Z = 512$. Detailed simulation parameters are listed in Table 1.

5.2. Result Analysis. We compare the proposed 3D REM-guided UAV path planning scheme (REM-guided scheme) with the one without REM (Without REM scheme). The simulator result in Figure 4 shows that the proposed REM-guided scheme assists the UAV to effectively avoid interference, and the average SINR exceeds 10 dB. However, the average SINR of the Without REM scheme is only 5.51 dB, and even in some locations the SINR is lower than −20 dB, which means the UAV loses communication with the base station.

To evaluate the performance of the proposed algorithm, we compare it with the IFDS model with fixed coefficients (Fixed coefficients scheme) and the IFDS model optimized based on the DDPG algorithm (DDPG-based scheme) [19]. And we test the three schemes in two environments, static path planning with global REM and real-time dynamic path planning with local REMs.

In the static environment, there are static obstacles of spheres, cylinders, and cones on the ground, and static radio interference abstracted as spheres. And the edge server distributes a compressed global REM to the UAV for global path planning.

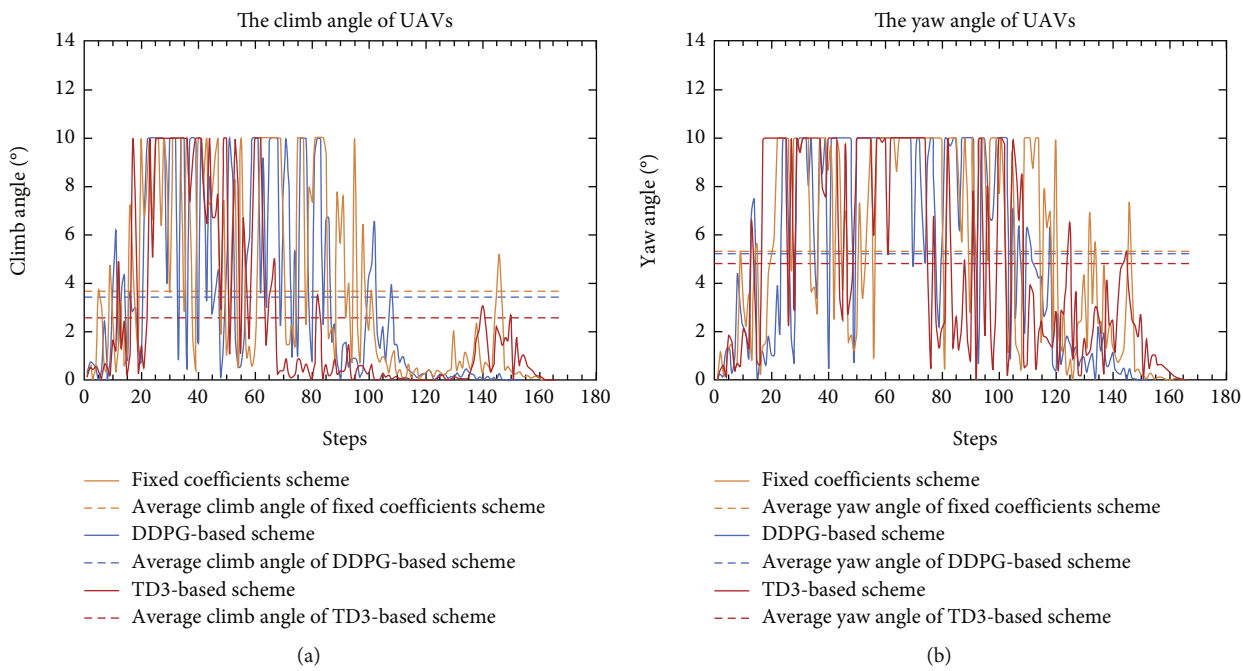


Figure 8: The climb and yaw angles of the three path planning schemes. (a) The climb angle of UAVs. (b) The yaw angle of UAVs.

As shown in Figure 5, all three schemes can plan a collision-free path for the UAV based on the global REM. However, since the Fixed coefficients scheme does not optimize the reaction coefficients of the IFDS model, the planned path is conservative, and the path length is 15.9 km. The planned paths of the DDPG-based scheme and the proposed TD3-based scheme partially overlap, but the DDPG-based scheme cannot find the optimal combination of coefficients when the UAV avoids obstacles. The path length of DDPG-based scheme is 14.8 km, and the path length of proposed scheme is 14.3 km.

We compare the climb and yaw angles of the three schemes in Figure 6. Since the UAV is subject to kinematic constraints, the climb change angle $|\alpha_t - \alpha_{t-1}|$ and the yaw change angle $|\beta_t - \beta_{t-1}|$ of the UAVs do not exceed the maximum constraint angle 10° during flight. The average climb change angle and the average yaw change angle of the Fixed coefficients scheme are 3.68° and 6.33°, and the average climb change angle and the average yaw change angle of the DDPG-based scheme are 3.22° and 6.04°. The average climb and average yaw change angles of the proposed TD3-based scheme are 3.18° and 4.44°. The proposed algorithm has the smallest changes in the climb angle and the yaw angle during the UAV flight, which means that the energy consumed is relatively less.

In a dynamic environment, we set the radio interference as dynamic interference and distribute local REMs in real-time for the UAV to avoid obstacles and interferences.

As shown in Figure 7(a), when the UAVs of the three schemes encounter the first dynamic interference sphere, the three UAVs choose different directions to avoid the interference. Then they encounter the second interference sphere in Figure 7(b) (step = 80), the DDPG-based and TD3-based schemes choose to fly from the bottom of the interference sphere, and the Fixed coefficients scheme flies from the left. From the overall view in Figure 7(c), the three schemes can effectively avoid obstacles and interference. However, the Fixed coefficients scheme conservatively avoids obstacles and interferences, which keeps a far safe distance. The two DRL schemes optimize the path length and avoid obstacles and interference at the same time through learning. The path length of the Fixed coefficients scheme is 17.3 km, the path length of the DDPG-based scheme is 16.7 km, and the path length of the proposed TD3-based scheme is 15.5 km, which is the shortest path of the three schemes.

We also compare the climb and yaw angles of the three schemes in dynamic environment in Figure 8. The average climb change angle and the average yaw change angle of the Fixed coefficients scheme are 3.68° and 5.32°, and the average climb change angle and the average yaw change angle of the DDPG-based scheme are 3.44° and 5.25°. The average climb and average yaw change angles of the proposed TD3-based scheme are 2.56° and 4.82°. The proposed algorithm has the smallest changes in the climb angle and the yaw angle in both static and dynamic environment.

The path planning and obstacle avoidance capabilities of the proposed algorithm are tested by distributing global and local REMs to UAVs in static and dynamic environments, respectively. Compared with the DDPG-based scheme and the Fixed coefficients scheme, the proposed scheme has the shortest path, while the climb angle and the yaw angle change minimally. The simulator results prove that the proposed REM-guided path planning scheme can effectively deal with the complex radio environment under communication connectivity constraints.

6. Conclusions

In the complex radio environment, due to the limitation of sensors carried by UAVs, the ability to perceive the environment is limited, and it is impossible to effectively avoid complex geographical obstacles and radio interference. In view of this, we proposed a 3D REM-guided path planning method for UAVs, which distributes compressed global REMs and detailed local REMs to UAVs to improve their awareness of the radio environment. An IFDS model is deployed on UAVs to allow them to avoid obstacles and plan paths. We proposed a TD3-based algorithm to optimize the reaction coefficients of the IFDS model. And the simulation results show that the proposed algorithm can effectively avoid static obstacles and dynamic interference under communication connectivity constraints and significantly improve the communication stability with a higher receive signal SINR and reduce the cost of UAV performing tasks with the shortest path.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 62171449, 62001483, and U19B2024.

References

[1] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," in *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[2] K. Katagiri and T. Fujii, "Mesh-clustering-based radio maps construction for autonomous distributed networks," in *2021 Twelfth International Conference on Ubiquitous and Future Networks (ICUFN)*, pp. 345–349, Jeju Island, Republic of Korea, August 2021.

[3] O. Esrafilian, R. Gangula, and D. Gesbert, "3D-map assisted UAV trajectory design under cellular connectivity constraints," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, Dublin, Ireland, June 2020.

[4] Q. Chen, H. Zhu, L. Yang, X. Chen, S. Pollin, and E. Vinogradov, "Edge computing assisted autonomous flight for UAV: synergies between vision and communications," *IEEE Communications Magazine*, vol. 59, no. 1, pp. 28–33, 2021.

[5] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, and M. Caccamo, "UAV path planning using global and local map information with deep reinforcement learning," in *2021 20th International Conference on Advanced Robotics (ICAR)*, pp. 539–546, Ljubljana, Slovenia, December 2021.

[6] J. Wu, H. Wang, N. Li et al., "Distributed trajectory optimization for multiple solar-powered UAVs target tracking in urban environment by adaptive grasshopper optimization algorithm," *Aerospace Science and Technology*, vol. 70, no. 1, pp. 497–510, 2017.

[7] L. Li, Q. Gu, and L. Liu, "Research on path planning algorithm for multi-UAV maritime targets search based on genetic algorithm," in *2020 IEEE International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, pp. 840–843, Chongqing, China, November 2020.

[8] Z. Ali, Z. Han, and B. Wang, "Cooperative path planning of multiple UAVs by using max-min ant colony optimization along with cauchy mutant operator," *Fluctuation and Noise Letters*, vol. 20, no. 1, p. 2150002, 2021.

[9] X. Wang, Z. Ning, S. Guo, M. Wen, and H. V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3225–3238, 2021.

[10] L. Yafei, W. Anping, C. Qingyang, and W. Yujie, "An improved UAV path planning method based on RRT-APF hybrid strategy," in *2020 5th International Conference on Automation, Control and Robotics Engineering (CACRE)*, pp. 81–86, Dalian, China, September 2020.

[11] J. Seo, Y. Kim, S. Kim, and A. Tsourdos, "Collision avoidance strategies for unmanned aerial vehicles in formation flight," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 6, pp. 2718–2734, 2017.

[12] A. Cahyadi, E. Darmawan, W. Dewanto, and I. Ardiyanto, "Application of artificial potential field for coverage control with collision avoidance under sensing constraints," *Journal of Control, Automation and Electrical Systems*, vol. 31, no. 2, pp. 304–318, 2020.

[13] H. Wang, W. Lyu, P. Yao, X. Liang, and C. Liu, "Three-dimensional path planning for unmanned aerial vehicle based on interfered fluid dynamical system," *Chinese Journal of Aeronautics*, vol. 28, no. 1, pp. 229–239, 2015.

[14] Y. Wang, H. Wang, J. Wen, Y. Lun, and J. Wu, "Obstacle avoidance of UAV based on neural networks and interfered fluid dynamical system," in *2020 3rd International Conference on Unmanned Systems (ICUS)*, pp. 1066–1071, Harbin, China, November 2020.

[15] J. Fan, Z. Wang, J. Ren, Y. Lu, and Y. Liu, "UAV online path planning technology based on deep reinforcement learning," in *2020 Chinese Automation Congress (CAC)*, pp. 5382–5386, Shanghai, China, November 2020.

[16] J. Wu, H. Wang, Y. Liu, M. Zhang, and T. Wu, "Learning-based fixed-wing UAV reactive maneuver control for obstacle avoidance," *Aerospace Science and Technology*, vol. 126, p. 107623, 2022.

[17] Z. Ning, S. Sun, X. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[18] V. -P. Chowdappa, C. Botella, J. J. Samper-Zapater, and R. J. Martinez, "Distributed radio map reconstruction for 5G automotive," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 2, pp. 36–49, 2018.

[19] J. Wu, H. Wang, Y. Wang, and Y. Liu, "UAV reactive interfered fluid path planning," *Acta Automatica Sinica*, vol. 47, no. 1, pp. 1–16, 2021.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6909314, 12 pages
https://doi.org/10.1155/2022/6909314

Research Article

MANet: End-to-End Learning for Point Cloud Based on Robust Pointpillar and Multiattention

Xingli Gan,[1] Hao Shi,[1] Shan Yang,[2] Yao Xiao,[3] and Lu Sun[4]

[1]*School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou 310023, China*
[2]*China Unicom Smart City Research Institute, Beijing 100048, China*
[3]*College of Sericulture, Textile and Biomass Sciences, Southwest University, China*
[4]*Department of Communication Engineering, Institute of Information Science Technology, Dalian Maritime University, China*

Correspondence should be addressed to Hao Shi; 222008855022@zust.edu.cn

Received 11 July 2022; Accepted 22 August 2022; Published 14 September 2022

Academic Editor: Xiaojie Wang

Detecting 3D objects in a crowd remains a challenging problem since the cars and pedestrians often gather together and occlude each other in the real world. The Pointpillar is the leader in 3D object detection, its detection process is simple, and the detection speed is fast. Due to the use of maxpooling in the Voxel Feature Encode (VFE) stage to extract global features, the fine-grained features will disappear, resulting in insufficient feature expression ability in the feature pyramid network (FPN) stage, so the object detection of small targets is not accurate enough. This paper proposes to improve the detection effect of networks in complex environments by integrating attention mechanisms and the Pointpillar. In the VFE stage of the model, the mixed-attention module (HA) was added to retain the spatial structure information of the point cloud to the greatest extent from the three perspectives: local space, global space, and points. The Convolutional Block Attention Module (CBAM) was embedded in FPN to mine the deep information of pseudoimages. The experiments based on the KITTI dataset demonstrated our method had better performance than other state-of-the-art single-stage algorithms. Compared with another model, in crowd scenes, the mean average precision (mAP) under the bird's-eye view (BEV) detection benchmark increased from 59.20% of Pointpillar and 66.19% of TANet to 69.91 of ours, the mAP under the 3D detection benchmark was increased from 62% of TANet to 65.11% of ours, and the detection speed only dropped from 13.1 fps of Pointpillar to 12.8 fps of ours.

1. Introduction

Autonomous driving uses sensors to detect and track moving objects, such as cars, pedestrians, and cyclists in real-time. Lidar is arguably the most important. Point cloud generated by lidar provide geometric structure information of objects and high-precision spatial coordinates, so how to use that information is extremely crucial [1].

With the outstanding achievements of computer vision and deep learning methods in pictures, extensive literature thinks about how to design end-to-end network results for point clouds. Unlike images represented as regular dense grids, 3D point cloud is not only irregular and disordered but also has the characteristics of uneven density and different shape and scaling ratio due to input-output size and order differences. Therefore, previous convolutional neural network (CNN) with regular grids is not suitable for point cloud. The method to solve this problem is to divide the space into regular geometry, such as $3 * 3 * 3$ cm and then manually design the feature extraction method. However, different feature extraction methods need to be designed for different environments and different detection targets, which is lack generality. In order to solve this problem, based on the PointNet designed by Qi et al. [2], an end-to-end detection network VoxelNet is proposed. VoxelNet [3] divides the point cloud into equidistant 3D voxels and encodes each voxel through the stacked VFE layer, and then, 3D convolution further aggregates the local voxel features to convert the point cloud into high dimensional volumetric representation. Finally, RPN produces test results. While the VoxelNet performance is strong, at 4.4 Hz, the inference time is too slow to deploy in real time. SECOND [4, 5]

improved the inference speed of VoxelNet but the 3D convolutions remain a bottleneck. Lang et al. [6] use a novel encoder that learns features on pillars (vertical columns) of the point cloud instead of voxel to predict 3D oriented boxes for objects which is highly efficient to compute due to the key operations can be formulated as 2D convolutions and Pointpillar runs at 72 Hz which has the obvious speed advantage. Although Pointpillar enables a trade-off between speed and accuracy, the performance is still unsatisfactory in challenging cases. As shown in Figure 1, the first row shows the corresponding 2D image. The second row demonstrates the 3D detection results produced by Pointpillar. Pedestrians were not detected due to the severe occlusion. We reveal the intrinsic reason that the key parameter in Voxel Feature Encode (VFE) is the size of the voxel. A coarser voxel leads to a smaller feature map and faster inference speed but has inferior performance, especially for small objects.

To solve this problem, TANet [7] introduced attention in the feature extraction stage and also divided FPN-RPN into coarse extraction and fine extraction to effectively solve the occlusion problem. Inspired by the words of TANet, we introduce the mixed-attention network (MA) and the Convolutional Block Attention Module (CBAM) [8]. MA combines channel-wise, point-wise, and voxel-wise to enhance key information and to suppress unstable points. Channel-wise is used to determine which channels in each voxel; point-wise is used to determine which points in a voxel; voxel-wise is used to determine which grids are more important in all voxel grids. The CBAM consists of two complementary attention modules: spatial attention and channel attention. It can assign more weight to the unshaded part and less weight to the shaded part. By inserting CBAM into FPN, more refined features are obtained to improve the accuracy of classification and regression.

The contributions of this paper include the following three aspects. Firstly, we introduce one novel single-stage framework, named MANet, which strikes a balance between accuracy and speed. Secondly, we introduce the MA model which can feedback the original features of the point cloud more effectively and retain better geometric properties of the point cloud. Thirdly, our model runs at 11 frames per second while achieving competitive performance on the KITTI dataset.

The rest of the paper is organized as follows. Section 2 introduces the related achievements on the 3D object detection and analyzes their merit and demerit. Section 3 presents network architecture and the elaboration of mixed attention and CBAM. Section 4 presented the relevant parameter setting of the experiment, evaluation criteria, and comparative results with other models. Section 5 concludes this paper.

2. Relation Work

2.1. 3D Object Detection. Object detection of the point cloud is an integral part of the 3D vision. Like the task of 2D target detection, 3D target detection is to locate all interested targets in a given scene accurately. At present, 3D object detection can generally be divided into region proposal-based and single-shot methods [5]. The methods based on the candidate region firstly predict the region with possible objects (also known as the proposal), then extract the features of each region to determine the object category of each candidate region. Figure 2 shows the development process of the 3D object detection algorithm. According to the different methods for generating proposals, these methods can be further divided into three categories: multi-view-based methods, segmentation-based methods, and point-based methods. Among them, Frustum-PointNet [9], a technique based on the PointNet++ [10] cone proposed by qi, achieves high benchmark performance, but its multistage design makes end-to-end learning impractical. The methods based on single-shot methods predict category probability directly and use a single-level network to regression the 3D bounding box of objects. These methods do not require region proposals and postprocess, making them ideal for real-time applications. According to the type of input data, the methods can be divided into three types: methods based on BEV (projection graph) [3, 11, 12], methods based on discretization, and methods based on the point. VoxelNet is the first method to deploy a point network in a lidar point cloud for target detection. The author transforms the irregular point cloud into regular voxels, then processes them by a set of three-dimensional convolution layers, followed by a two-dimensional backbone and a detection head. This makes end-to-end learning possible, but like earlier work that relied on 3D convolution, requiring 225 ms of reasoning time (4.4 Hz) for a single point cloud. Pointpillar proposes converting voxels into pseudoimages and then using mature 2D target detection methods to carry out detection, which successfully realized real-time reasoning. TANet researches the relationship between point cloud, space, and voxel based on the Pointpillar, suppressed the unimportant and highlighted the important parts through the attention mechanism, and solves the interference of background points to a certain extent. HVNet [13] solves the balance problem of accuracy and speed caused by voxel division by integrating multiscale voxels.

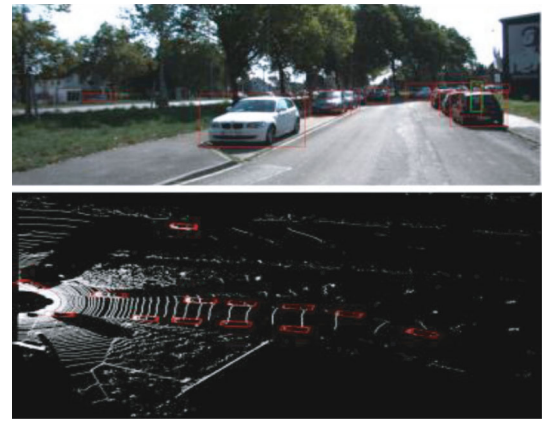


Figure 1: Detection results for Pointpillar.

2.2. Attention. In recent years, attention has gained popularity as a plug-and-play module for the existing basic convolutional neural network (CNN) architecture [13–17].

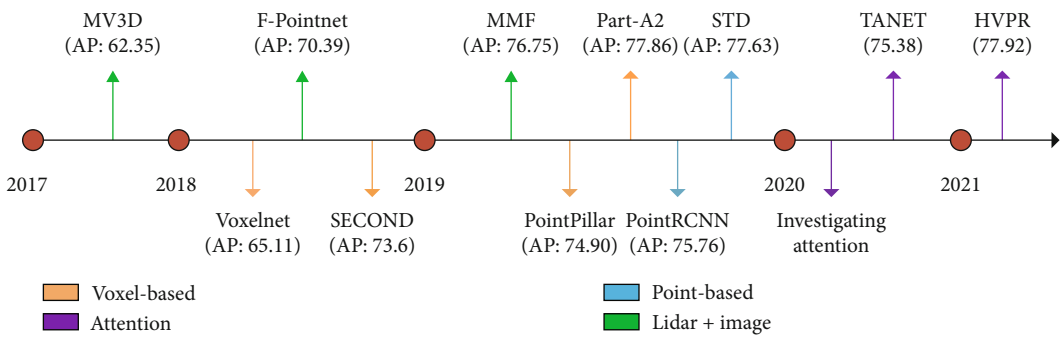


Figure 2: The development process of a 3D object detection algorithm based on point cloud representation.

Attention mechanisms are aimed at mimicking the human visual system by focusing on more relevant features to the target rather than an entire scene containing some unrelated background. Many methods have been introduced to estimate attention (weight) maps to reweight the original feature maps learned from CNN. SENet uses the global average set feature to calculate channel-level attention in their squeeze and excitation module for image-related tasks. They ignore spatial attention, which plays an essential role in deciding "where" as shown in [18]. After analyzing the defects of SENet and SKNet [19], a lightweight module CBAM is proposed to realize the mixing of space and channel by serial instead of parallel, which dramatically reduces the running time. Some scholars have tried to introduce attention to point cloud network architecture in recent years. The experiment of [20] verified that both 2D and 3D attention modules could be inserted into the existing modules to improve the feature extraction capability of the point cloud network. TANet combines channel attention, point attention, and voxel attention to enhance the critical information of the target and suppress unstable points, thus improving the robustness of the network. In addition, [21–23] introduce transformer into point cloud classification to enhance it by four points over PointNet with half the number of parameters.

2.3. Datasets. Semantic 3D is a large-scale point cloud classification benchmark, which provides a 3D point cloud dataset of natural scenes with large labels, totaling over 4 billion points and 8 category labels. And it also covers a wide variety of urban scenarios. KITTI [24] can not only have lidar, image, GPS, and INS data but also have manually labeled segmentation tracking results, which can be used to objectively evaluate the effect and performance of a large-range of 3D modeling and fine classification. The 3D object detection benchmark consists of 7,481 training images and 7,518 test images along with the corresponding point clouds, including a total of 80,256 labeled objects. The recent H3D dataset [25] records the crowded and highly interactive urban scenes, including a total of 1 million labeled instances in 27721 frames. The KAIST multispectral dataset [26] is a multispectral pedestrian detection dataset, which provides black-and-white thermal imaging image pairs during the day and night. Through the complementary advantages of color image and thermal imaging, the dataset improves the accuracy of pedestrian detection and overcomes the problems of previous pedestrian detection data, such as blocked pedestrians, messy background, and unclear imaging at night. Other noteworthy multimodal datasets include [27] providing driving behavior tags, [28] providing location classification tags, and raw data without semantic tags. The nuScenes dataset [29] contains 3D bounding box of 23 classes and 8 properties, with his annotation number being more than one times KITTI 7, resolving errors due to data enhancement.

3. Approach

This part introduces our object detection network based on attention object networks shown in Figure 3, called mixed-attention-Pp. The model structure diagram can be divided into VFE, multiscale pseudograph feature learning module, and detection head. Firstly, the original point cloud is transformed into a grid composed of voxels and obtains a more discriminative representation through mixed attention. Then, the features are aggregated by maximum pooling and finally dispersed back into a pseudoimage of $H * W * C$. In the feature extraction stage, we added CBAM to the original FPN [30–33]. The ability to capture detail is improved by inserting a CBAM module in the upsampling stage. Finally, a single shot multibox detector (SSD) is used to detect the position and classify categories.

We defined some common variables of the point cloud in advance. Point cloud set $P = \{p_i = [x_i, y_i, z_i, r_i]^t \in \mathrm{R}\}_{i=1,2,\cdots,M}$ where x_i, y_i, and z_i represent the coordinates of the midpoint in lidar space. r_i represents other spatial features, such as reflectivity and normal. M represents the number of point clouds. We use $(c_x, c_y, c_z, h, w, \ell, \theta)$ to define a 3D bounding box where c_x, c_y, and c_z represent the center point of the box; h, w, and ℓ represent the size of the box; and θ represents the direction of motion of the object.

3.1. Stacked Mixed-Attention. The entire range of space S is discretized into the point cloud P, where the range of P is (W^*, H^*, D^*). P is equally divided into a specific voxel grid $V = \{v^1, \ldots, v^k\}$ where $v^k \in R^{N*C}$, k represents the index of voxels, N represents the maximum number of point clouds per voxel, and c represents the characteristic number of point clouds. Each grid size is $w = W^*/v_w^*$, $h = H^*/v_h^*$, and

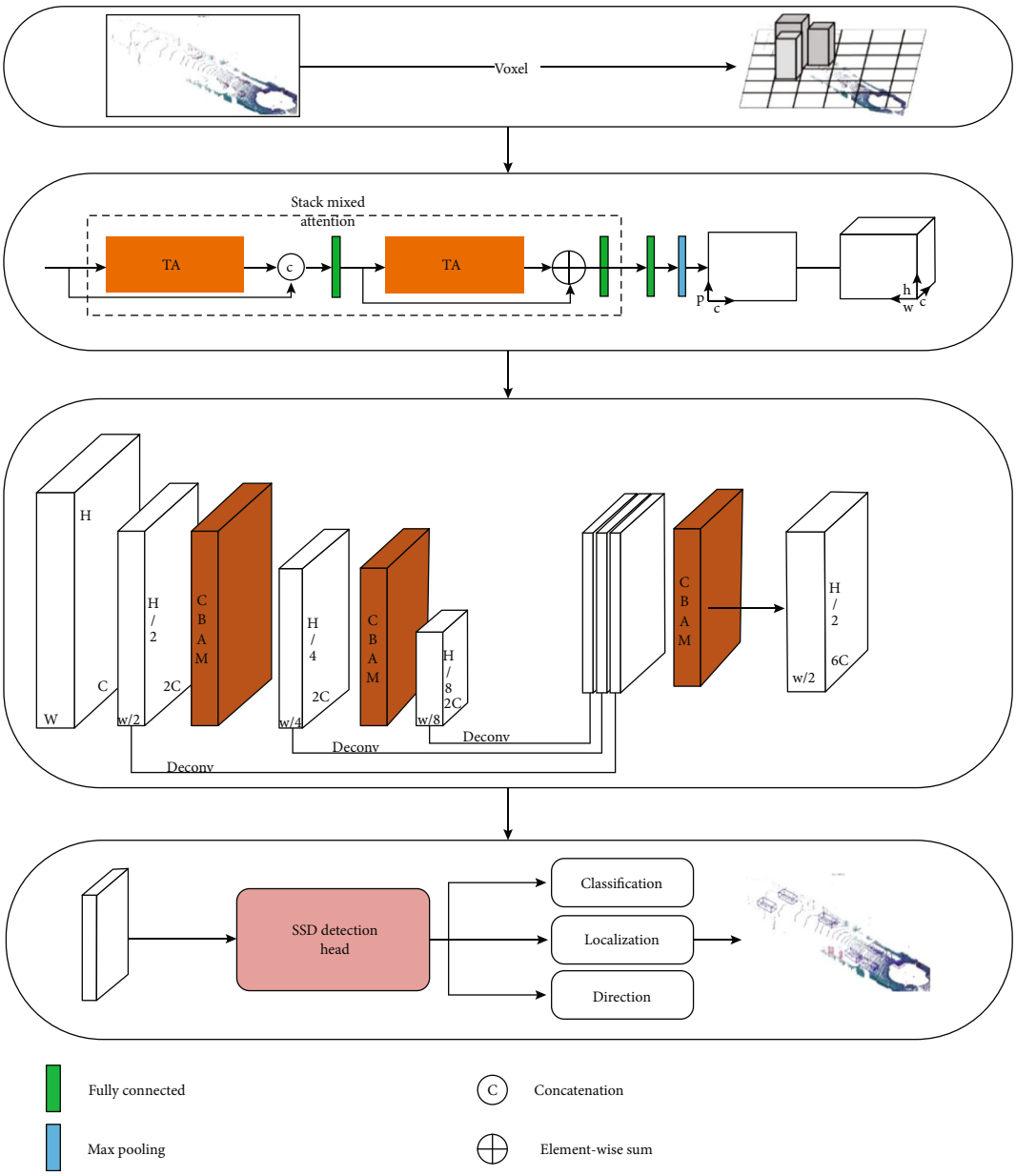


Figure 3: The full pipeline of network.

$D = D^*/v_d^*$. In the z-axis, we regard it as a whole, so the D is 1. They are shown in Figure 4.

3.1.1. Point-Wise Attention. Given the input v^k where its shape is (K, N, C). K represents the number of voxels, N represents the number of voxels, and C represents the number of channels. Firstly, we use maxpooling to make the feature transfer from the previous layer a vector then we use two MLP to obtain global coding features S^k.

$$S^k = W_2\delta\left(W_1E^K\right), \tag{1}$$

where E^K is point-wise, W_1 and W_2 are the weight parameters of two MLP, respectively, and δ is the ReLU activation function.

3.1.2. Channel-Wise Attention. Channel-wise attention is very similar to channel-wise attention. The only difference is maxpooling in the first dimension of input v^k. Specifically, we do maxpooling to convert the feature map of the previous layer into vector $U^k \in R^{1*C}$, which aggregates the features of all points on each channel. Then, through the two MLP to estimate the attention characteristic map $T^k = W_2'\delta(W_1'(U^k))$ where $W_2' \in R^{R*C}$, $W_1' \in R^{R*C}$. Then, S^k and T^k are combined through multiply, and the attention weight M^k is obtained through sigmoid activation function. Finally, F_1^K is obtained by dot product of v^k and M^k.

$$M^k = \delta\left(S^k * T^k\right). \tag{2}$$

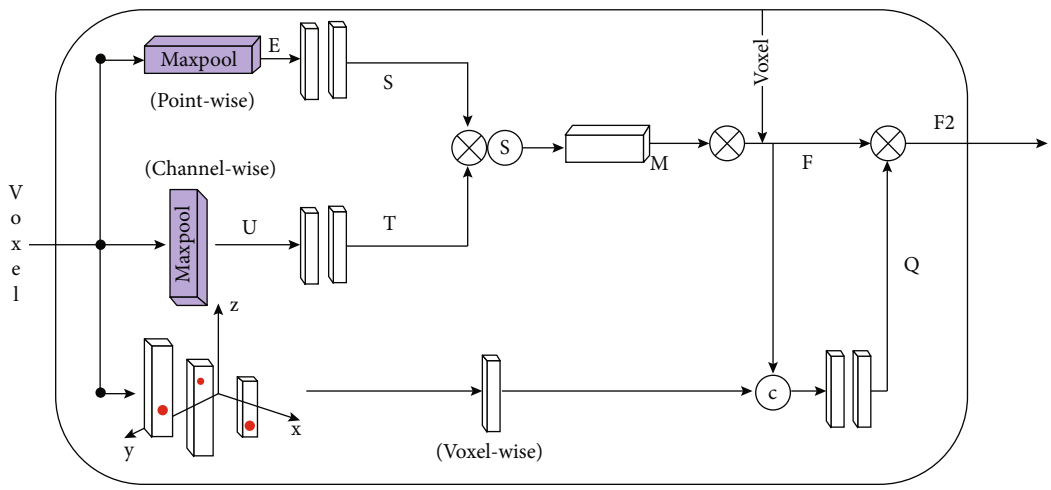


Figure 4: The architecture of the mixed attention module.

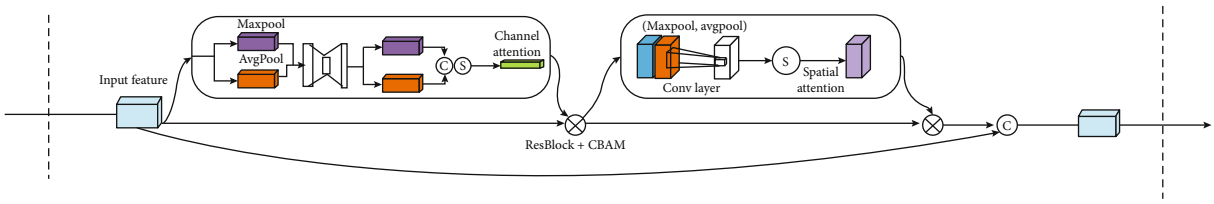


Figure 5: The architecture of CBAM.

3.1.3. Voxel-Wise Attention. Further, the importance of voxels is judged by focusing on voxels. The input for voxel-wise is V_C and F_1^K where V_C represents the center of gravity point of each 3D voxel grid. Its size is $(K, 1, 3)$. We first expand V_C to $(K, N, 3)$ in the first dimension and then concatenate V_C and F_1^K in the first dimension and then obtain voxel level attention weight Q which size is $(K, 1, 1)$ through two full connection layer and sigmoid. Finally, $F_2^K = q_k * F_1^K$.

3.1.4. Stack-TA. Considering that TA module directly acts on point cloud, it does not contain high-dimensional semantic features. The generalization ability of this network is insufficient. We chose to stack TA in order. We choose to stack TA in a sequential manner. Specifically, VX and VC are input into the first TA, and the output is M. M is used as the input of the second TA attention. The difference from the former is that we directly add the input and output instead of splicing. Finally, maxpooling is used to aggregate the features of all voxels.

3.2. Convolutional Block Attention Module (CBAM). As shown in Figure 5, CBAM is the mixed-attention composed of two continuous attention blocks. To be specific, with the intermediate feature graph $F \in R^{C*H*W}$ as input. CBAM successively deduced the one-dimensional channel attention graph $M_C \in R^{C*1*1}$ and the 2D space attention graph $M_s \in R^{1*w*h}$. The whole attention process can be summarized as follows: graph $M_s \in R^{1*w*h}$. The whole attention process can be summarized as follows:

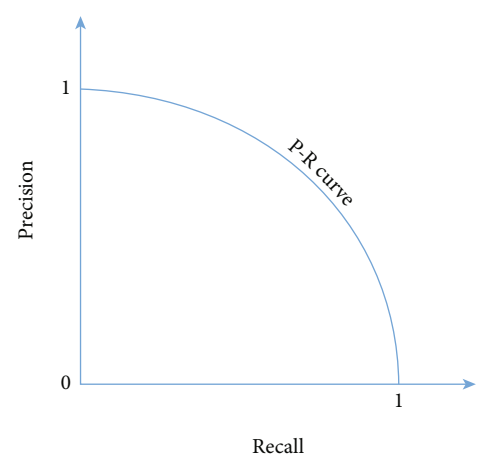


Figure 6: The P-R curve.

$$F' = M_C(F) \otimes F,$$
$$F'F' = M_S\left(F'\right) \otimes F'. \tag{3}$$

3.3. Backbone. We used a trunk similar to FPN [31], whose structure is shown in Figure 3. The backbone network can be divided into three parts. The one on the left is the

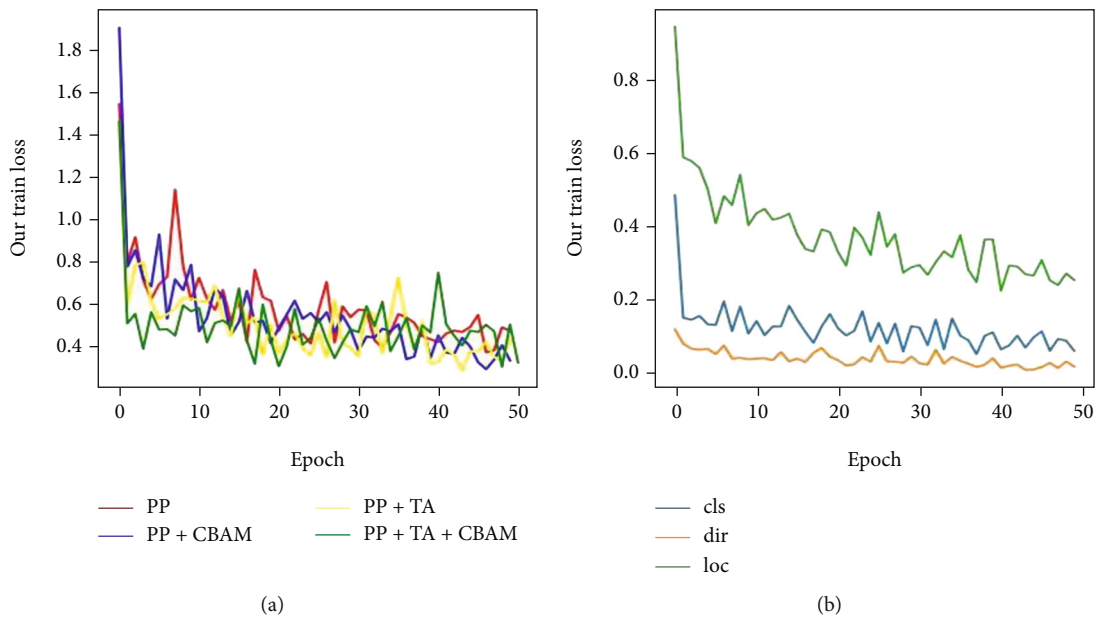


Figure 7: The loss of train. (a) The ablation experiments of different attention. (b) The component of the total loss.

downsampling network, which produces features with smaller and smaller spatial resolutions. The CBAM module in the middle inputs the feature map into CBAM after convolution, and CBAM extracts useful information while inhibiting irrelevant information. On the right is the upsampling module, which combines the feature layer $(6c, w/2, h/2)$ through deconvolution. The final output features are a concatenation of all features that originated from different strides.

3.4. Detection Head and Loss. We use the same detection header as SSD [5, 34] to locate 3D objects. Ground truth (gt) and anchors are defined by $(x, y, z, w, l, h, \theta)$ where (x, y, z), (w, l, h), and θ are the center point, size, and angle of the box, respectively. Local residuals between the anchors and the ground truth are defined:

$$\begin{aligned} \Delta x &= \frac{x^{gt} - x^{\alpha}}{d}, \\ \Delta y &= \frac{x^{gt} - x^{\alpha}}{d}, \\ \Delta z &= \frac{z^{gt} - z^{\alpha}}{h^{\alpha}}, \\ \Delta w &= \log \frac{w^{gt}}{w^{\alpha}}, \\ \log \frac{l^{gt}}{l^{\alpha}} \Delta h &= \log \frac{h^{gt}}{h^{\alpha}}, \\ \Delta\theta &= \sin\left(\theta^{gt} - \theta^{\alpha}\right), \end{aligned} \tag{4}$$

where $d = \sqrt{(w^a)^2 + (l^a)^2}$ and gt and a are, respectively, the ground truth and anchor box. In order to train our model, we carried out regression, classification and memory update, the total loss is

$$\mathscr{L} = \frac{1}{N_{pos}} \begin{pmatrix} \lambda_{reg}\mathscr{L}_{reg} + \lambda_{dir}\mathscr{L}_{dir} \\ +\lambda_{cls}\mathscr{L}_{cls} + \lambda_{mem}\mathscr{L}_{mem} \end{pmatrix}, \tag{5}$$

where N_{pos} represents positive anchors; the equilibrium parameter of the corresponding loss and the total localization loss is defined as follows:

$$\mathscr{L}_{reg} = \sum_{r \in (\mathscr{X}, \mathscr{Y}, \mathscr{Z}, \mathscr{W}, \mathscr{l}, \mathscr{h}, \theta)} \text{Smooth}L1(\Delta \mathfrak{r}). \tag{6}$$

Table 1: Ablation experiments on the effect of CBAM, SE, and TA as well as different combination settings.

Methods	Car	Pedestrian	Cyclist	mAP
Baseline [6]	74.9	43.5	64.5	59.0
SE [18]	68.3	44.94	51.72	54.9
CBAM [8]	76.1	50.2	64.3	63.5
TA [7]	78.8	46.47	59.60	62
TA + CBAM	78.8	51.74	64.74	65.1

4. Experiment and Discussion

4.1. Parameter Settings. We set the threshold on the XY of the scene of the point cloud to (0, 70.4) and (-40, 40). The resolution of the pillar is 0.16 m, the maximum number of columns is 12000, and the maximum number of points per column is 100. If the points are less than 32, then we randomly sample the points in the column. If the points' number is more than 100, then we use farthest point sampling for downsampling. We share the matching strategy for the box and prediction box as Pointpillar. Specifically, the anchors consist of the

Table 2: Results on the KITTI test BEV detection benchmark.

Methods	3D mAP	Car			Pedestrian			Cyclist		
		Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
VoxelNet [3]	58.25	89.25	79.26	77.39	46.13	40.74	38.11	66.70	54.76	50.55
Pp [6]	66.19	88.35	86.10	79.83	58.6	50.23	47.19	79.14	62.25	56.00
F-P [9]	65.39	88.70	84.00	75.33	58	50.22	47.20	75.38	61.96	54.98
PIXOR [12]	N/A	89.38	87.30	77.97						
MV3D [38]	N/A	86.02	76.90	68.49						
SECOND [39]	60.56	88.07	79.37	77.95	55.1	46.27	44.76	73.67	56.04	48.78
MANet	69.91	89.21	86.36	83.10	61.4	55.01	51.23	82.81	68.36	63.76

Table 3: Results on the KITTI test 3D detection benchmark.

Methods	Mod	Bev mAP	Car			Pedestrian			Cyclist		
			Easy	Mod	Hard	Easy	Mod	Hard	Easy	Mod	Hard
MV3D [38]	Lidar&img	N/A	71.09	62.35	55.12						
F-P [9]		57.35	81.20	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39
VoxelNet [3]	Lidar	49.05	77.47	65.11	57.73	39.48	33.69	31.5	61.22	48.36	44.37
SECOND [39]		56.69	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	46.90
Pp [6]		59.20	79.05	74.99	68.30	52.08	43.53	41.49	75.78	59.07	52.92
TANet [7]		62	83.81	75.38	67.66	54.92	46.67	38.63	73.93	59.60	53.59
MANet		65.11	83.47	78.85	71.89	56.02	51.74	45.58	80.08	64.74	60.79

following 7 parameters (c_x, c_y, c_z, h, w, l, θ), and the anchor direction, θ, is applied to two orientations (0 and $\pi/2$). We regard anchors with intersection over union (IOU) greater than 0.6 as positive samples and those less than 0.2 as negative samples. We ignore those anchors with IOU between (0.2, 0.6) when calculating the loss. We select the nonmax suppression (NMS) score of 0.5 in the postprogress step. Due to the large body size gap between cars and people, we set the corresponding parameters for different objects.

Car. Thresholds are set to (0, 70.4) and (-40, 40), the size of the prior box is set to (1.6, 30.9, 1.5), and the size of the confidence interval was set to (0.45, 6).

Pedestrian and Cyclist. Since pedestrians are blocked and the number of point clouds is sparse, we set the scene range to (0, 48) and (-20, 20), the size of the prior box is set to (0.6, 0.8, 1.73), and the confidence interval to (0.2, 0.6). The prior size of all boxes is set to (1.6, 3.9, 1.5) and (0.6, 0.8, 1.73) for pedestrians and individuals. The confidence interval for the vehicle is (0.45, 0.6), and the pedestrian is (0.5, 0.35).

4.2. Sample Ground Truths from the Database. This paper takes the KITTI dataset as the benchmark. Since the dataset samples are only 7361 frames and the number is small, we can manually add some objects to the point cloud to improve the effect of model training. This paper adopts the same data enhancement method as Pointpillar. Firstly, points in the prior box are removed and recorded from the training set. Secondly, N samples are randomly selected, and the prior box and point cloud of the selected samples were randomly rotated (- $\pi/20$, $\pi/20$) and translated (0, 0.25); then, we added the samples to the training set.

Table 4: Results on the test nuScenes 3D detection benchmark.

Methods	Car	Pedestrian	Cyclist	mAP
Baseline [6]	62.5	50.2	64.5	57.6
PVRCNN	64.8	46.7	—	—
Centerpoint	66.1	62.4	67.6	65.3
Point augmenting	62.2	64.6	73.3	66.7
Ours	65.1	63.7	65.9	64.9

4.3. Algorithm Performance Evaluation. In the field of target detection [35–37], recall and precision are mainly used as the performance measure of the algorithm. Precision (P) and recall (R) are, respectively, defined as follows:

$$R = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \quad P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{7}$$

where TP, FP, TN, and FN are represented as true and false positive and true and false negative examples, respectively. It can be seen that the denominator of P is the total number of boxes detected by the detec-tor, and the denominator of R is the total number of boxes given by GT. Since R and P are a pair of paradoxical quantities. To balance them, the "P-R" curve obtained with P as the vertical axis and R as the horizontal axis is used to reflect the relationship in the Figure 6.

The average accuracy comes from the PR curve. In practice, we do not directly calculate the PR curve, but instead smooth the PR curve. That is, for each point on the PR

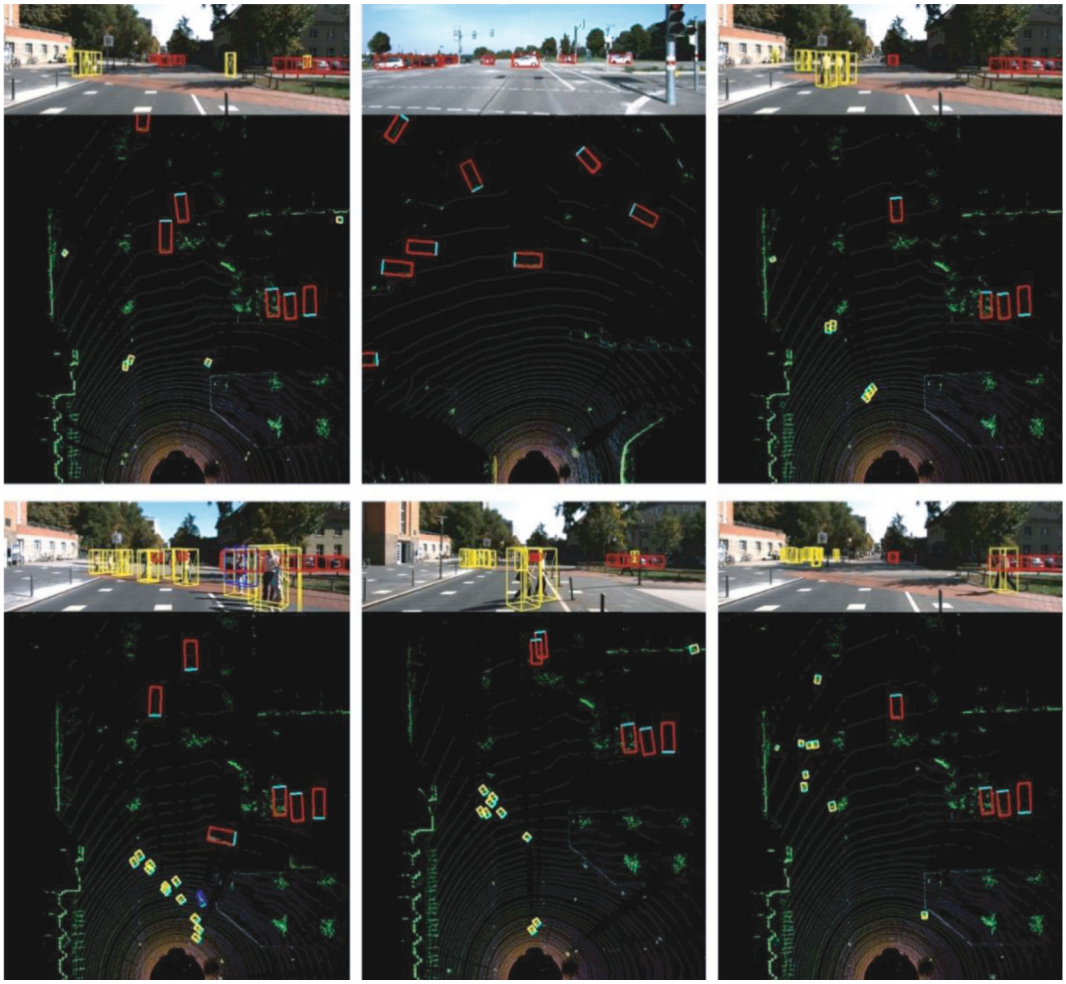


Figure 8: Results of 3D detection on the KITTI test set. For better visualization, the 3D boxes detected using lidar are projected onto images from the left camera.

curve, the value of precision takes the value of the largest precision on the right side of that point. In addition, for a better performance of the reaction model, we also introduced mAP (mean average precision) to represent the mean of AP values for all types.

$$\mathrm{mAP} = \frac{\left(\sum_{i=1}^{C} \mathrm{AP}_i\right)}{C}. \tag{8}$$

In this paper, according to the complexity of the environment, we divided the objects into easy, moderate, and hard and counted the models for 3D bounding box AP, Bev AP, Bev mAP, and Bev bounding box, respectively.

4.4. Performance Analysis

4.4.1. Loss. Train runtime is measured on a GTX 1050 Ti GPU. As mentioned in 3.4, loss consists of three parts: classification, localization, and direction. Figure 7(b) shows the changes of loss in the training process. As can be seen in the figure, the model directly generates the category probability and position coordinate value of the object without generating the candidate region first and then classifying the candidate region, so it has a faster detection speed. In Figure 7(a), we made statistics on the decrease of loss in 4 situations: Pointpillar, Pointpillar +CBAM, Pointpillar+TA, and Pointpillar +TA + CBAM during the training process. The results in Figure 7(a) show that whether inserting CBAM or TA, the loss value of final convergence is lower than the original Pointpillar.

4.4.2. Analysis of the Attention Mechanisms. Table 1 presents ablation studies of the proposed attention mechanisms. The recall for cars, pedestrians, and cyclists is set to (0.7, 0.5, 0.5), and the score threshold is set to 0.4. We removed both the TA and the CBAM from our model as the baseline and achieve a 3D mAP of 59.0%; with only CBAM and SENET, we can see that if only the channel attention mechanism is added, the accuracy is reduced by KITTI 4% while CBAM outperforms the baseline model by 4.5% from Table 1. This suggests that the spatial information is beneficial for the regression of the 3D bounding box. We add the TA module to the baseline base, and the performance was promoted to 62. Finally, we reinserted TANet and CBAM into the model,

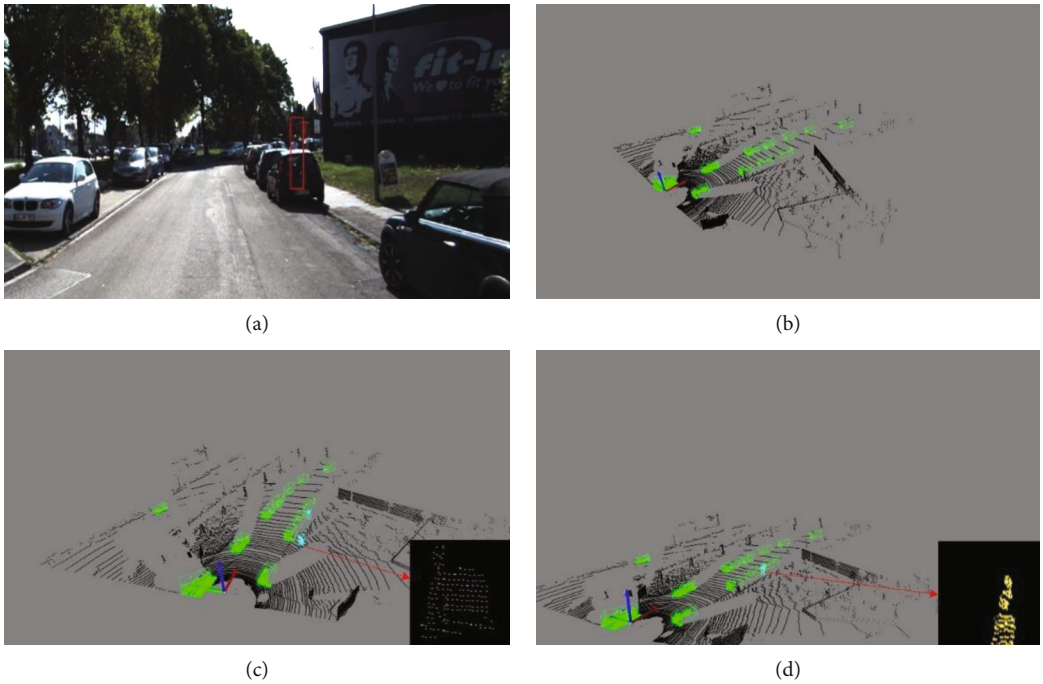


Figure 9: Results of 3D detection on the KITTI test set: (a) the corresponding 2D image; (b) the detection results of the Pointpillar; (c) the detection results of the Pointpillar+SE; (d) the detection results of ours.

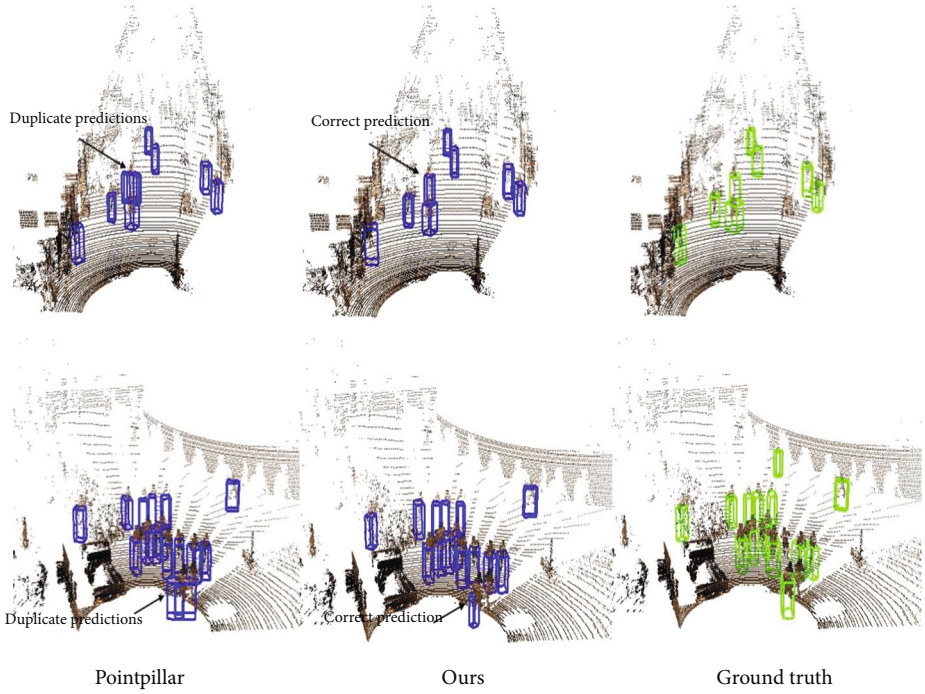


Figure 10: Results of 3D detection on the nuScenes test set. The first column is Pointpillar, the second column is our algorithm, and the third is the ground truth.

with the model accuracy further improved to 65. The experiments show that the performance of the network can be effectively improved by assigning weights to different points for some objects with occlusion. Figure 7 shows the loss of train. Figure 7(a) shows the ablation experiments of different attention. Figure 7(b) shows the component of the total loss.

4.4.3. Quantitative Analysis. In Tables 2 and 3, we compared our model with the others. For comparison convenience, the car, pedestrian, and cyclists' recall are set to (0, 7, 0.5, 0.5), and the score threshold is set to 0.1. All detection results are measured using the official KITTI evaluation detection metrics which are bird's-eye view (BEV) and 3D. We

classified the 3D detection model as lidar and lidar and image-based. As can be seen from the table, our mAP rose from Pointpillar at 66.19% and 59.20%, respectively, to 69.91 and 65.1; the 3D bounding box in difficult environments has increased by 4%, which is an exciting result. This suggests that adding attention modules makes the network still work in the face of complex environments. In Table 4, we also add the latest detection model-TANet; we all adopt TA attention in the VFE stage; in the pseudoimage feature extraction stage, our model adopts FPN + CBAM mode, and TANet adopts CFR (coarse regression module and a fine regression); the results show that our model outperforms TANet. We also tested on the large dataset nuScenes, and the experimental results are shown in Table 4. Our method achieves some improvement in single-stage centerpoint compared to pillar and PVRCNN, but point augment achieves high accuracy in cycling and pedestrian detection results. The reason is that point augment uses a multimodal approach, which integrates the semantic information of images, and pictures have a natural advantage for capturing small objects.

4.5. Test Result. Figure 8 shows some test results on the KITTI test set to visually show the detection effect of the model. Each image consists of 2 parts: the first row is the predicted 3D bounding box projected into the image, and the second row is the predicted results of Bev where the red represents the car, yellow is the cyclist, and blue is the pedestrian. The images in the first row select the road with few pedestrians and no occlusion. It can be seen that our algorithm detects all objects without occlusion, and the second row selects the scene when there are many pedestrians. It can be seen that pedestrians will block each other and vehicles, and our algorithm can still have a high recall rate.

To further verify the robustness of our algorithm, we compared our method with other current state-of-the-art algorithms. In Figure 9(a), most of the pedestrian's body is covered by the vehicle, and the radar has collected less than 30 points. The original Pointpillar experiment results are shown in Figure 9(b), and the pedestrians were not found. The Pointpillar+SE algorithm (Figure 9(c)) mistakenly detects the road signs as pedestrians. We analyze that the fake image weakens the spatial feature and strengthens the channel feature. SE attention further strengthens the channel feature, so there will be a problem of false detection. While we propose a mixed-attention model that the global spatial information, local spatial information and point features are integrated so the pseudoimage contains more spatial features. CBAM combines space with channels, so it can show excellent performance. In addition to comparing the effect on KITTI, we also verified it in nuScenes. As shown in Figure 10, our algorithm detected the sparse pedestrians missed by second and avoid the repeated detection problem caused by crowding.

5. Conclusion

In this paper, we have designed the MA-CBAM-Pp for single-stage 3D object detection, which improves detection performance, particularly in crowd scenes. Two attention models are inserted into Pointpillar, where the mixed attention is added in VFE which feedback the original features of the point cloud more effectively and retain better geometric properties of the point cloud, and the CBAM attention module was embedded in FPN to mine the deep information of pseudoimages from spatial dimension and channel dimension. Significantly, our result on KITTI with MA-CBAM-Pointpillar outperforms the previously best result that uses TANet by about 3.11%. Detailed experimental comparisons have demonstrated the value of our method, which improves detection accuracy by a large margin in occlusion scenarios. Meanwhile, the visualization results also demonstrate that our inserted modules help improve the accuracy of single-stage 3D object detection. Future work will do fine regression based on this model to further improve the accuracy of the model.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflict of interest.

Authors' Contributions

All authors participated in the discussion of the initial design, methodology, and derivation of the method presented in this paper. X.G. and H.S. designed, conceived the simulation experiments, and analyzed the data. H.S. and D.F.D. performed the training, analyzed the results, and wrote the paper. Z.H.H. and Y.S. reviewed the paper, and all authors participated in amending the manuscript. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China (project: Underground space disaster information perception and emergency communication equipment) (No. 2020YFC1511705), the National Natural Science Foundation of China (62101088), and the Radar Signal Processing National Defense Science and Technology Key Laboratory Fund (6142401200101).

Supplementary Materials

The pictures in the paper are provided. *(Supplementary Materials)*

References

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The Kitti vision benchmark suite," in *the 2012*

IEEE conference on computer vision and pattern recognition, pp. 3354–3361, Providence, RI, USA, 2012.

[2] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: deep learning on point sets for 3D classification and segmentation," in *the Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, Hawaii,USA, 2017.

[3] Y. Zhou and O. Tuzel, "VoxelNet: end-to-end learning for point cloud based 3D object detection," in *the Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4490–4499, Salt Lake City, USA, 2018.

[4] S. Ji, X. Wei, M. Yang, and Y. Kai, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[5] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *the Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4203–4212, Salt Lake City, USA, 2018.

[6] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: fast encoders for object detection from point clouds," in *at the Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12697–12705, Long Beach,USA, 2019.

[7] Z. Liu, X. Zhao, T. Huang, R. Hu, Z. Yu, and X. Bai, "Tanet: robust 3D object detection from point clouds with triple attention," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 7, pp. 11677–11684, 2020.

[8] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: convolutional block attention module," in *the Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

[9] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3D object detection from RGB-D data," in *the Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 918–927, Salt Lake City, USA, 2018.

[10] C. R. Qi, L. Yi, S. Hao, and L. J. Guibas, "PointNet++: deep hierarchical feature learning on point sets in a metric space," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[11] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, Madrid, Spain, 2018.

[12] B. Yang, W. Luo, and R. Urtasun, "Pixor: real-time 3D object detection from point clouds," in *the Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7652–7660, Salt Lake City, USA, 2018.

[13] M. Ye, S. Xu, and T. Cao, "HVNet: hybrid voxel network for lidar based 3D object detection," in *the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1631–1640, Seattle,USA, 2020.

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model," 2016, https://arxiv.org/abs/1602.07360.

[15] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, https://arxiv.org/abs/1409.1556.

[17] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-V4, inception-resnet and the impact of residual connections on learning," in *the Thirty-first AAAI conference on artificial intelligence*, San Francisco, USA, 2017.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, Salt Lake City, USA, 2018.

[19] W. Wu, Y. Zhang, D. Wang, and Y. Lei, "SK-Net: deep learning on point cloud via end-to-end discovery of spatial keypoints," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 4, pp. 6422–6429, 2020.

[20] S. Qiu, W. Yunfan, S. Anwar, and C. Li, "Investigating attention mechanism in 3D point cloud object detection," in *2021 International Conference on 3D Vision (3DV)*, pp. 403–412, London, United Kingdom, 2021.

[21] M.-H. Guo, J.-X. Cai, Z.-N. Liu, M. Tai-Jiang, R. R. Martin, and S.-M. Hu, "Pct: point cloud transformer," *Computational Visual Media*, vol. 7, no. 2, pp. 187–199, 2021.

[22] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7794–7803, Salt Lake City, USA, 2018.

[23] H. Zhao, L. Jiang, J. Jia, P. H. S. Torr, and V. Koltun, "Point transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268, 2021.

[24] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: the Kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[25] A. Patil, S. Malla, H. Gang, and Y.-T. Chen, "The H3D dataset for full-surround 3D multi-object detection and tracking in crowded urban scenes," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9552–9557, Montreal, QC, Canada, 2019.

[26] Y. Choi, N. Kim, S. Hwang et al., "Kaist multi-spectral day/night data set for autonomous and assisted driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 934–948, 2018.

[27] Y. Chen, J. Wang, J. Li et al., "Lidar-video driving dataset: learning driving policies effectively," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5870–5878, Salt Lake City, USA, 2018.

[28] M. Brossard, A. Barrau, and S. Bonnabel, "AI-IMU dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.

[29] H. Caesar, V. Bankiti, A. H. Lang et al., "nuScenes: a multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, Seattle,USA, 2020.

[30] Z. Ding, H. Xu, and M. Niethammer, "VoteNet: a deep learning label fusion method for multi-atlas segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 202–210, Cham, 2019.

[31] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, Hawaii,USA, 2017.

[32] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 770–779, Long Beach,USA, 2019.

[33] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "PointASNL: robust point clouds processing using nonlocal neural networks with adaptive sampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5589–5598, Seattle,USA, 2020.

[34] Z. Yang, Y. Sun, S. Liu, and J. Jia, "3DSSD: point-based 3D single stage object detector," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11040–11048, Seattle,USA, 2020.

[35] F. Nobis, E. Shafiei, P. Karle, J. Betz, and M. Lienkamp, "Radar voxel fusion for 3D object detection," *Applied Sciences*, vol. 11, no. 12, p. 5598, 2021.

[36] J. Zhang, J. Wang, X. Da, and Y. Li, "HcNet: a point cloud object detection network based on height and channel attention," *Remote Sensing*, vol. 13, no. 24, p. 5071, 2021.

[37] W. Zheng, H. Xie, Y. Chen, J. Roh, and H. Shin, "PIFNet: 3D object detection using joint image and point cloud features for autonomous driving," *Applied Sciences*, vol. 12, no. 7, p. 3686, 2022.

[38] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1907–1915, 2017.

[39] Y. Yan, Y. Mao, and B. Li, "Second: sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 3323239, 14 pages
https://doi.org/10.1155/2022/3323239

Research Article

Fault Diagnosis to Nuclear Power Plant System Based on Time-Series Convolution Neural Network

XianLing Li,[1] **DongJiang Han**,[2] **XinFa Dai**,[2] **ShuYu Lv**,[2,3] **Mo Tao**,[1,4] **Wei Zheng**,[1] **and YiBin Tang**[2]

[1]*Science and Technology on Thermal Energy and Power Laboratory, Wuhan, Hubei 430205, China*
[2]*Wuhan Digital Engineering Institute, Wuhan Hubei 430074, China*
[3]*Harbin Engineering University, Harbin Heilongjiang 150001, China*
[4]*School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China*

Correspondence should be addressed to ShuYu Lv; lvsy@hrbeu.edu.cn and Mo Tao; taomo@buaa.edu.cn

Received 27 June 2022; Revised 8 August 2022; Accepted 17 August 2022; Published 5 September 2022

Academic Editor: Xiaojie Wang

Nuclear power plant (NPP) is a highly complex engineering system which has typical internal feedback and strong component coupling. With these features, most NPP systems have high risk of radioactive release, which makes it essential to perform fault detection (FD) to the NPP systems. To address this challenge, this paper proposes a FD mechanism named characteristic time-series convolutional neural network (CT-CNN) based on principal component analysis (PCA), time-series analysis, and convolutional neural network (CNN) mechanisms. First, the models of NPP FD system are formulated. Then, the PCA mechanism is applied to extract the features of the NPP system. Next, the time-series analysis and CNN approaches are applied to realize FD to the NPP system. With the above mechanisms, the proposed approach has not only shown strong stability and become adaptive to different data set, but also preserves both time and state characteristics of the NPP system. In experiment, it shows the proposed approach can achieve better performance in both detection accuracy and variance than the classic back propagation, LSTM method, and standard CNN algorithms. More significantly, its optimal accuracy can be as high as 99.8%.

1. Introduction

The structure of industrial nuclear power plant (NPP) control system is rather complex as it consists of many interconnected systems and equipment [1]. Due to the high complexity, the internal feedback phenomenon is obvious in NPP system [2], e.g., the change of coolant temperature affects the coolant volume, and this effect further puts pressures to the steam generator, etc. In other words, any small deviation in the NPP system may cause a failure quickly and then makes this failure spread throughout the whole NPP system, which can consequently result in disasters [3]. Therefore, the fault detection (FD) to the NPP system is of great significance.

Since FD technology can ensure the safety and reliability of the NPP system, this technology has been extensively studied in recent years. Currently, the related studies on FD can be divided into two kinds: model-based methods and model-free methods. Model-based methods commonly rely on ideal assumptions and physical knowledge to establish mathematical models, e.g., the differential equation model based on thermodynamic equation and nuclear reactor point dynamics equation can simulate the internal state of NPP to a certain extent through numerical operation and residual evaluation. However, in practice, the model-based method cannot establish a sufficiently accurate mathematical model, especially for highly complex coupling systems like NPP. The idealized assumptions are likely to

deviate from the actual situation, and the numerical operation is likely to delay the valuable diagnosis time. These problems limit the application of this method in many application scenarios.

Compared with model-based methods, model-free methods are more popularly applied. This method is mainly divided into expert system methods and data-driven methods. The expert system models represented by fault tree analysis (FTA) [4] and random forest (RF) [4] have achieved good results. However, the expert system also faces some problems in the practical applications, such as the impossibility of exhaustive knowledge base caused by the limitation of expert knowledge and the contradiction between the poor adaptability and expert knowledge caused by the limitation of empirical knowledge. Different from the expert system, the data-driven method is more flexible and concise [5]. It does not need to carry out essential theoretical analysis and empirical rule summary, and it only needs to collect the historical operation data of NPP to establish a relatively feasible diagnosis model.

Since the data-driven approaches show the above advantages, many data-driven methods have been proposed for the FD of NPP system, such as k-nearest neighbor (KNN) method [6], support vector machine (SVM) [7], principal component analysis (PCA) [8], and other classical statistical machine learning methods. These methods are effective and have achieved good performance. However, they have the problems of poor anti-interference ability, low recognition accuracy, and high time complexity. The determined mapping function and linear classifier make many approaches unable to use the massive operation state data to extract the state features and summarize the system experience of NPP. Improved classic back propagation network [9], deep belief network (DBN) [10], and recurrent neural network (RNN) [11] are applied to this problem, but these methods still have the problem of unstable training.

To improve the FD performance, many approaches try to combine the concepts of different FD methods. Yao et al. [12] proposed a full-range FD method based on state information imaging. With this method, the state information of NPP is expressed by gray image, and the image features are extracted by Kernel Principal Component Analysis (KPCA). Then, the FD is realized by using various classification methods. Peng et al. [13] proposed a method combining correlation analysis (CA) and deep belief network (DBN), in which CA was used to reduce the dimension and DBN was in charge of training and diagnosis. Wang et al. [7] realized the diagnosis of coolant circuit of NPP system by using SVM and improved particle swarm optimization (PSO). Li and Lin [14] used the convolutional neural network (CNN) algorithms to extract the characteristics of instantaneous data of NPP system and realized the diagnosis of eight types of faults. The above methods are effective in realizing FD of NPP, yet they still have the limitation in ignoring the time relationship of NPP system data as well as the relationship between time and state. To solve this problem, many studies use the time-series approach in the mechanical failure analysis [15]. Yao et al. [16] integrated the time-domain and frequency-domain characteristics of multichannel acoustic signals through CNN and realized the diagnosis of gear fault. Chen et al. [17] input the mechanical monitoring signal of rolling bearing into 1DCNN for training and obtained the FD model of rolling bearing.

With the time-series approaches, the FD performance can be improved. However, this approach cannot be applied directly to the NPP system, and this is because the NPP system has multiple state characteristics. To address this challenge, many new FD approaches based on CNN [18, 19] and long short-term memory (LSTM) network have been proposed for the NPP system. He et al. [20, 21] used Markov to process the multistate data. They transformed the data into color images after flatting the multistate data into one-dimensional data and extracted features through the CNN image processing functionality. The results showed that this approach could realize the diagnosis of eight types of faults of NPP system. Choi and Lee [11] combined the online monitoring technology based on signal reconstruction as well as the LSTM network to achieve FD of NPP system. However, most of these methods do not mine the time features well, or they destroy the continuity of the time series and eliminate the relationship between time series and multidimensional states. Moreover, less work has considered the relationship between NPP state data and faults from a global perspective.

To address the above challenges, this paper proposes a data-driven FD method based on time-series analysis. On the one hand, the method uses PCA method to reduce the dimension to exclude some irrelevant features and integrated some related features but not lose much information; on the other hand, the method considers the connection between the features and time series, arranged these data into a matrix, and used the convolution method to extract both partial features and partial time series at the same time, which preserves the structure compared with [20] so that it can realize the FD of NPP system more accurately and efficiently. The method does not have strict requirement on the setting of the parameter values, which makes it have high adaptability and can be applied to different NPP application scenarios.

The structure of this paper is as follows: Section 2 describes the problem of realizing FD in the NPP system; Section 3 builds the FD model and presents the solutions to find the optima of this model. In Section 4, the experiments are conducted and the results are discussed. Finally, in Section 5, a conclusion is drawn.

2. Problem Description

The operational status data of NPP system, which reflects the health status and fault information, is collected online in real time through the instrumentation and control system (I&C) system. This section gives the following definitions about the operational status data of NPP system.

Definition 1. Let $X = [x_1, x_2, x_3, \cdots, x_M] \in \mathbb{R}^M$. Then, $x_i (i \leq M)$ denotes the operation state data of NPP collected by the ith sensor at a certain time t, such as the water level and pressure of the pressurizer, the steam flow of the steam

generator, the opening and feedwater flow of the feedwater regulating valve, the water level, and the exhaust steam flow of the main condenser.

Definition 2. Let

$$P = \begin{bmatrix} X_{t_1} \\ X_{t_2} \\ \vdots \\ X_{t_T} \end{bmatrix} \in \mathbb{R}^{M\times T} \tag{1}$$

denote the operational status data set collected in time period $t_1 - t_T$, where $X_{t_j}(j \le T)$ is the state vector in Definition 1. Suppose the sampling time-interval of the sensors is equal, that is, for any j, there is $t_{j+1} - t_j \equiv h(h > 0)$, where h is the sampling period.

P is a time-series matrix which has the following features: ① the data of the row or column of the matrix represents the same characteristic, and ② the data are arranged in chronological order.

Definition 3. Let $E = \{e_1, e_2, e_3, \cdots, e_D\}$ denote the set of fault types, where $e_i(i \le D)$ denotes a certain type of fault, such as heat pipe water loss accident, cold pipe water loss accident, rupture of steam pipe inside the containment, rupture of steam pipe outside the containment, loss of water supply accident, and closing of the main steam isolation valve. e_0 indicates normal operational state, $E' = e_0 \cup E$.

The characteristic information of the fault is stored up not only in the operating state vector X_{t_j} at the time t_j, but also in the trend $X_{t_j}, X_{t_{j+1}}, X_{t_{j+2}}, \cdots$, e.g., when the pressure data of steam generator B drops sharply, there is a fault of steam line B pipe rupture. Therefore, this paper fully considers the time sequence characteristics and uses the sliding time window to intercept the time subsequence for FD.

Definition 4. Let

$$W = \begin{bmatrix} X_{t_l} \\ X_{t_{l+1}} \\ \cdots \\ X_{t_{l+w}} \end{bmatrix} \tag{2}$$

denote the sliding time window, where W is the continuous submatrix of P and the number of columns is equal, ℓ is the starting index of sliding window, and w is the size of sliding time window. Then, the sliding time window represents the fault state of time t_{l+w}. Supposing that the starting index of the sliding window increases ℓ each time, the number of sliding windows generated from a state data set P is $Q = \lfloor (T - w)/l + 1 \rfloor$.

The objective is to establish a fitting mapping $G : W \longrightarrow e_i$ that makes its output close to the real operational state of NPP as much as possible.

3. Model Design

3.1. Architecture Design of FD Model. To establish the mapping $G : W \longrightarrow e_i$, this paper designs a FD model based on the CNN feature extraction, PCA and sliding window mechanisms, as is shown in Figure 1. Firstly, PCA dimensionality reduction to the original data set is performed, and then the time-series submatrix is intercepted through the sliding window. Next, the CNN coder is used to analyze the features of the input sequence shaped by the sliding window. Finally, a small-size feature matrix is output to realize the classification of the input sequence.

The specific implementation process is shown in Figure 2, and the working process is as follows:

(1) Obtain the initial samples from NPP system and standardize the samples

(2) Establish the PCA model matrix to reduce the dimension, and then obtain the principal component state characteristics

(3) Reconstruct and reshape the principal component state characteristics by the sliding window method, and ensure the characteristics available to the input structure of CNN network

(4) Carry out convolutional operation and fuse time characteristics and state characteristics

(5) Pool operation and highlight key features

(6) Repeat Steps ④ and ⑤

(7) Flatten the output features

(8) Fully connect the output eigenvector, calculate the value of relevant neurons by forward propagation, and optimize the model according to the cross-entropy loss function

(9) Repeat ④ to ⑧ until the loss function reaches the target range

3.2. Data Preprocessing. Generally, the status data of NPP monitored by the I&C system cannot be directly used for the input data of the model. It is necessary to standardize the data based on different unit systems. Moreover, it is essential to reduce the dimension of high-dimensional status data and extract the fault features.

3.2.1. Data Standardization Processing. To realize the feature extraction of NPP operational state time series, this paper uses Z-score regularization method to convert the sample data to the same dimension. The standardized conversion formula of Z-score is as follows:

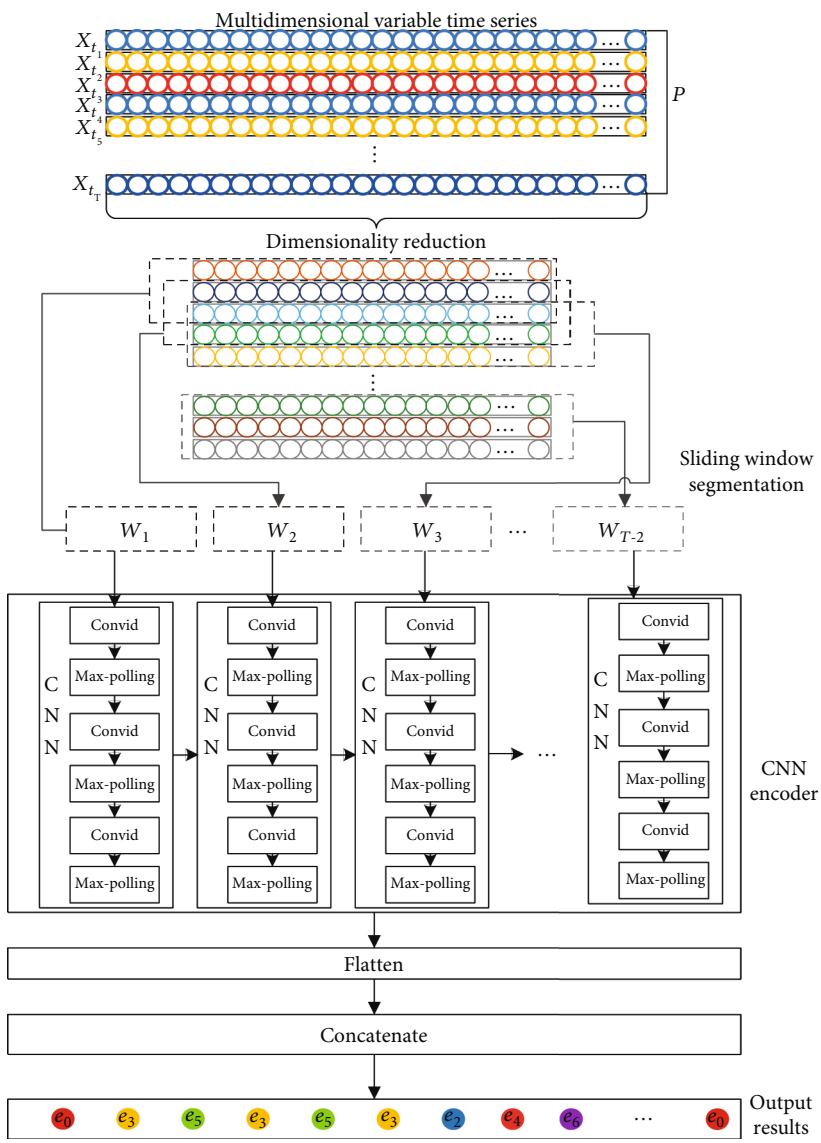


Figure 1: FD architecture for NPP system.

$$\hat{x}_i = \frac{x_{i,t_j} - \overline{x_i}}{s_i}, \tag{3}$$

where x_{i,t_j} denotes the state data collected by the *ith* sensor at time t_j, $\overline{x_i} = (1/T)\sum_{j=1}^{T} x_{i,t_j}$ denotes the average value of the *ith* state data in time period T, and $s_i = \sqrt{(1/(T-1))\sum_{j=1}^{T}(x_{it_j} - \overline{x_i})^2}$ denotes the standard deviation of x_{i,t_j}.

3.2.2. Data Dimensionality Reduction Processing. In terms of the coupling and correlation of the operational state data of NPP, the PCA method is used to reduce the dimension of the sample data.

The PCA method divides the state vector space $\mathbb{R}^M$ in which the sample data set is located into principal subspace and residual subspace. The principal subspace represents the change trend of the data, and the residual subspace represents the data disturbance. Then, any sample can be decomposed into the projection of two subspaces, and the projection difference between samples in the residual subspace is small. Therefore, it can ignore the projection difference in the residual space and select a vector to replace the projection of all samples in the residual subspace, so as to reduce the dimension.

Considering the orthogonal base $U = (u_1, u_2, u_3, \cdots, u_{M-1}, u_M)$, U can be spanned into an $\mathbb{R}^M$; then, the sample data set state vector $\hat{X}_{t_k}$ can be written as

$$\hat{X}_{t_k} = \sum_{i=1}^{M} a_{ki} u_i, \tag{4}$$

where $a_{ki} = \hat{X}_{t_k} u_i^T$.

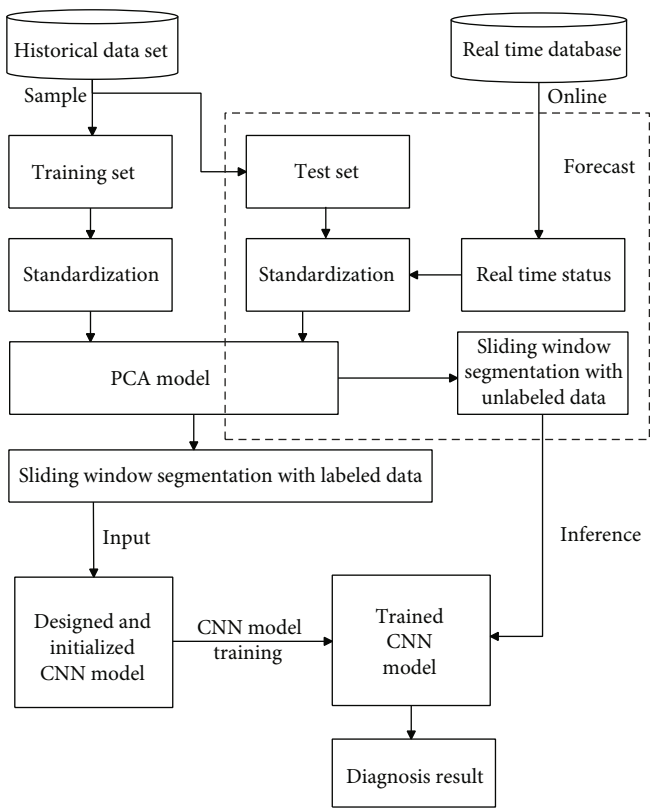


Figure 2: Flow chart of FD for NPP system.

Let $\tilde{X}_{t_k}$ denote the target vector of dimension reduction, which can be expressed by

$$\tilde{X}_{t_k} = \sum_{i=1}^{m} z_{ki} u_i + \sum_{i=m+1}^{M} b_i u_i, \tag{5}$$

where m is the dimension of the principal subspace, $\sum_{i=1}^{m} z_{ki} u_i$ is the linear representation of $\tilde{X}_{t_k}$ based on U in the principal subspace, and $\sum_{i=m+1}^{M} b_i u_i$ is the linear representation of $\tilde{X}_{t_k}$ based on U in the residual subspace.

Let J denote the objective function of dimension reduction, which can be expressed by

$$J = \frac{1}{M} \sum_{k=1}^{M} \left\| \hat{X}_{t_k} - \tilde{X}_{t_k} \right\|^2. \tag{6}$$

That is:

$$J = \frac{1}{M} \sum_{k=1}^{M} \left(\hat{X}_{t_k} - \tilde{X}_{t_k} \right) \left(\hat{X}_{t_k} - \tilde{X}_{t_k} \right)^T. \tag{7}$$

By solving the partial derivative of the coordinates of $\tilde{X}_{t_k}$, the minimum of J [22] can be denoted as

$$J = \sum_{i=m+1}^{M} u_i^T S u_i, \tag{8}$$

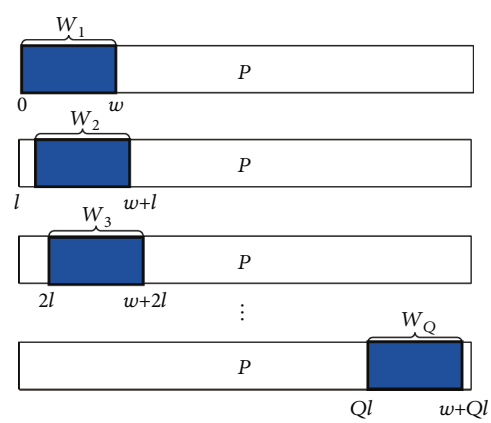


Figure 3: Process of generating sliding time window from the matrix P.

where $S = (1/M) \sum_{k=1}^{M} (\hat{X}_{t_k} - \bar{X}_{t_k})^T (\hat{X}_{t_k} - \bar{X}_{t_k})$ is the covariance matrix of the data vector.

u_i is linearly independent. Thus, when each $u_i^T S u_i$ takes minimum value, it can get the minimum value of J. Next, Lagrange multiplier method can be used for each $u_i^T S u_i$:

$$F(u_i) = u_i^T S u_i + \lambda_i \left(1 - u_i^T u_i\right). \tag{9}$$

The solution is

$$S u_i = \lambda_i u_i. \tag{10}$$

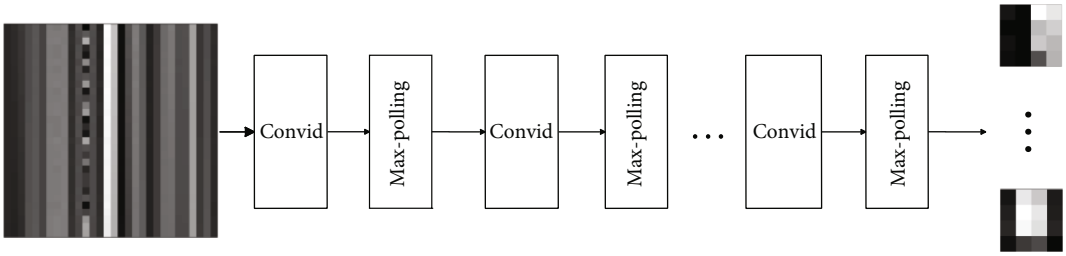


Figure 4: CNN network structure for feature extraction.

Time	Press containment	Temp average fuel	Temperature hot leg A	Temperature hot leg B	• • •	Volume CMT1	Volume CMT2
1	15.51325	300.83334	321.92672	321.92673	• • •	100	100
2	15.51380	300.83337	321.92679	321.92679	• • •	100	100
3	15.51336	300.83267	321.92670	321.92670	• • •	100	100
4	15.51222	300.83289	321.92657	321.92657	• • •	100	100
5	15.51096	300.83362	321.92667	321.92667	• • •	100	100
⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮
30	15.48050	300.84402	321.94049	321.94049	• • •	100	100
31	15.47944	300.84424	321.94077	321.94077	• • •	100	100
⋮	⋮	⋮	⋮	⋮	⋱	⋮	⋮
300	15.42608	300.85300	321.95016	321.95016		100	100

Figure 5: Random time-series faults used in the study case.

Equation (10) is to decompose the state sequence by singular value and solve the eigenvalue of the covariance matrix. After obtaining the eigenvalue and eigenvector, Equation (8) can be expressed as

$$J = \sum_{i=m+1}^{M} \lambda_i. \tag{11}$$

In order to minimize the objective function J, the minimum $M - m$ eigenvalues are used. In this case, the basis of the principal component subspace is the eigenvector corresponding to the maximum m eigenvalues. At this time, it can get a characteristic time-series set with m principal component state features from the original data set though the orthogonal transformation method.

3.3. Sliding Time Window Design. The process of generating W from time-series matrix P is shown in Figure 3.

The form of sliding window W depends on window size w and sliding step ℓ in terms of Definition 4: The smaller w is, the less information a single window W contains, and the less accurate and faster fault classification is. The larger ℓ is, the smaller Q value is, and the more accurate training speed is. This is because less information is contained in the time series and the reduced total amount of data set. Therefore, the specific value of W and ℓ needs to be adjusted according to the characteristics of the data set.

3.4. CNN Model Design. The key point to fit the mapping G is to obtain the state and time features of W. In this study, the CNN is used to extract the state and time features and then fuse these feature data. The network structure of CNN is shown in Figure 4.

The CNN consists of two parts: (1) The convolutional computation of feature extraction and (2) pool the characteristic matrix.

3.4.1. Step 1: Convolutional Computation. The convolutional computation extracts the features in the two dimensions through the local linear and nonlinear transformation of the time-series matrix. This process can be expressed by the following formula:

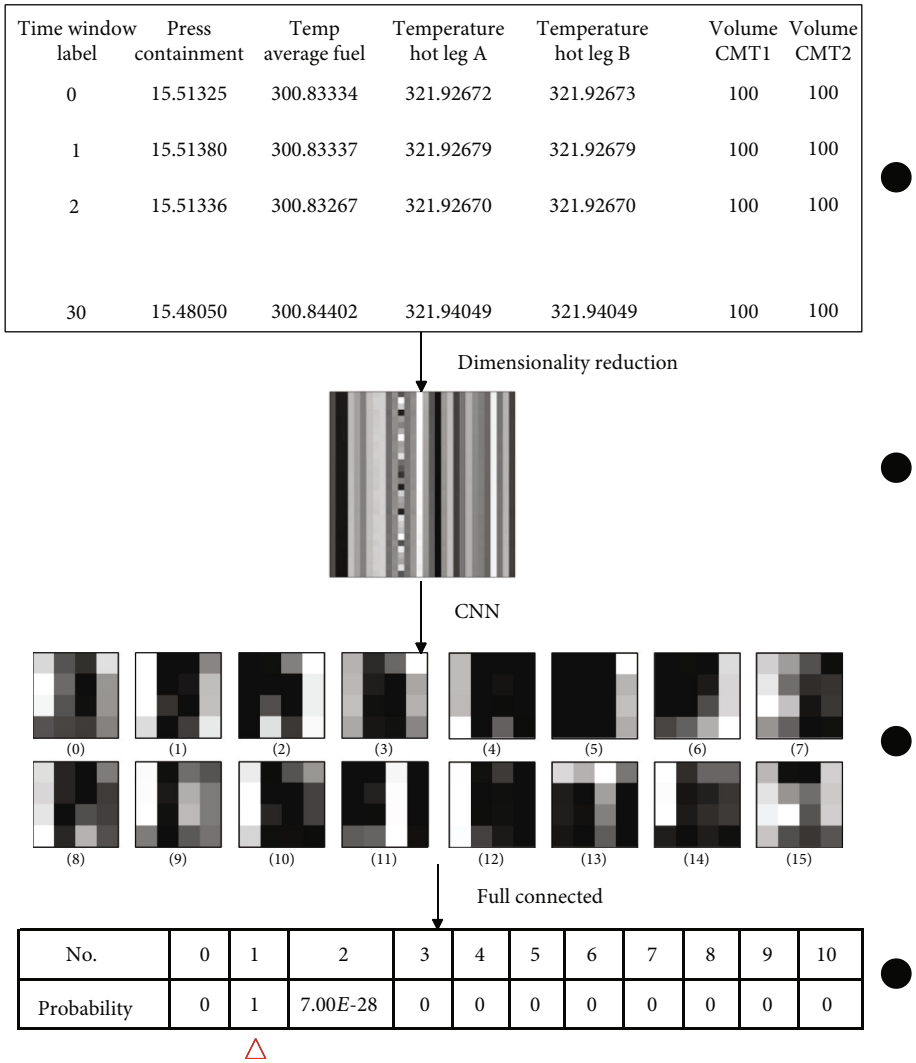


Figure 6: Fault classification process based on CNN.

$$z^{(r+1)}{}_{u,v} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} x^{(r)}{}_{i,j} \cdot k^{(r+1)}{}_{u-i,v-j}, \tag{12}$$

$$a^{(r+1)}{}_{u,v} = f\left(z^{(r+1)}{}_{u,v} + b^{(r+1)}\right). \tag{13}$$

Equation (12) represents the part of linear transformation of the input time-series matrix, where $z^{(r+1)}$ represents the linear transformation of the time-series matrix in the $(\mathrm{r}+1)_{\mathrm{th}}$ layer, $x^{(r)}$ represents the input of the time-series matrix of the r_{th} layer, and $k^{(r+1)}$ represents the coefficient matrix of the linear transformation in the $(\mathrm{r}+1)_{\mathrm{th}}$ layer.

Equation (13) represents the part of nonlinear transformation based on linear transformation, where matrix $b^{(r+1)}$ represents the offset added to the time-series matrix $z^{(r+1)}$, f represents an activation function, namely a nonlinear function which uses ReLu function, and $a^{(r+1)}$ represents the time-series matrix output.

3.4.2. Step 2: Pooling Process. The dimension of this matrix needs to be scaled after the convolutional process. Since the size of the time-series characteristic matrix is still the same as the original matrix, it is necessary to perform the pooling operation to make the network lightweight.

The pooling method in this study uses the maximum pooling mechanism. This method can perceive the small changes in the time series compared with other time series. After extracting time-series features via CNN, the number of parameters becomes less, and the interference such as noise becomes lower. This enables the speed of the model to be faster and the prediction of model to be more accurate.

The output of the time-series matrix of layer $\mathrm{r}+1$ through the pooling operation can be expressed as

$$x^{(r+1)}{}_{u,v} = \beta^{(r+1)} \cdot a^{(r+1)}{}_{i,j},$$
$$\beta^{(r+1)} = \begin{cases} 1 & a^{(r+1)}{}_{i,j} = \max\left(\partial a^{(r+1)}\right) \\ 0 & others \end{cases}. \tag{14}$$

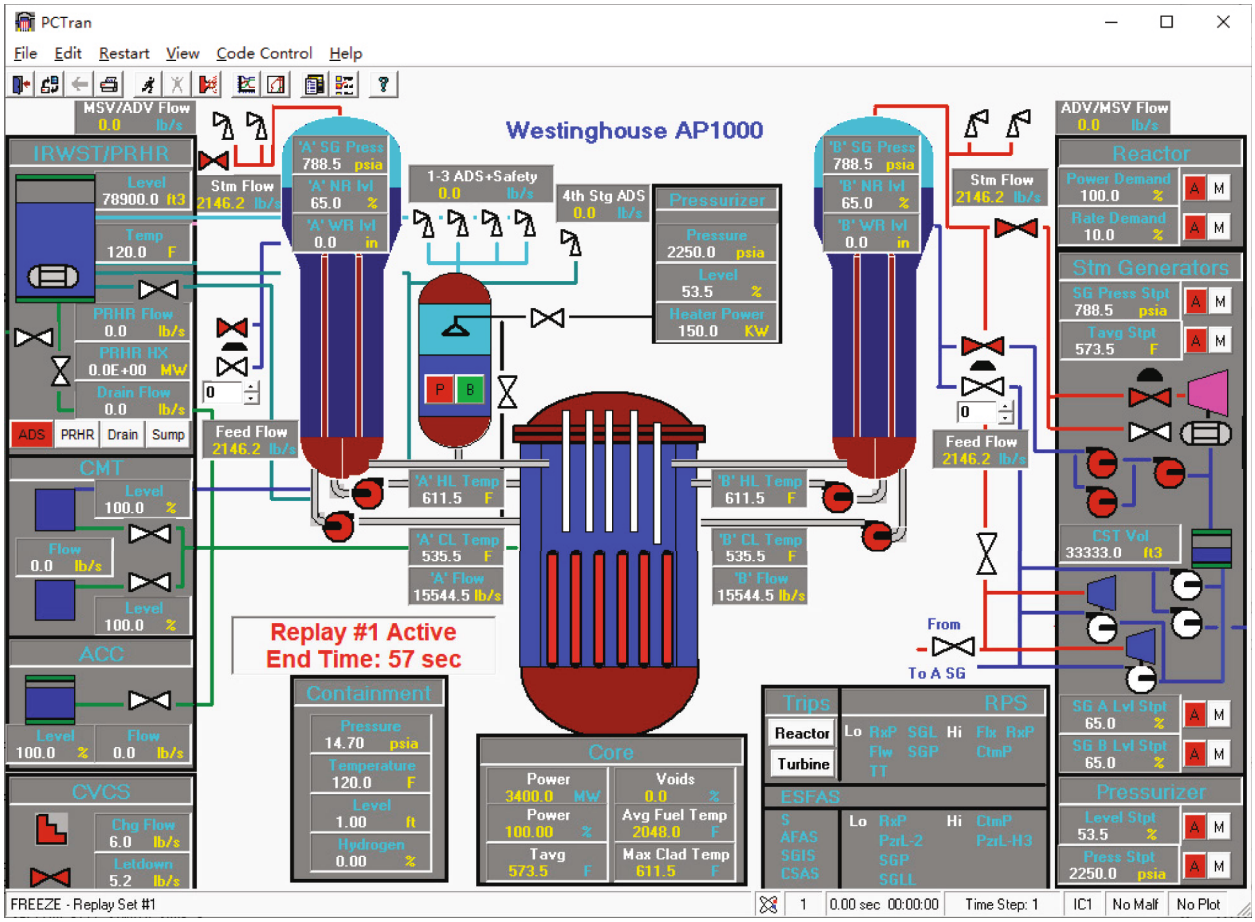


Figure 7: Pctran/AP1000 software for NPP simulation.

where $\partial a^{(r+1)}$ refers to the perception domain in the pooling process and the extraction of the maximum value of the perceptual submatrix domain reflects the feature of the maximum pooling.

3.4.3. Step 3: Full Connection. The full connection layer classifies the feature data extracted from the CNN layer and outputs the corresponding result, which can be expressed as

$$z^{(k+1)} = f\left(z^{(k)} W^{(k+1)} + b'^{(k+1)}\right), \tag{15}$$

where $z^{(k)}$ represents the one-dimensional data sequence input in the k layer. Specifically, when k is equal to 0, it represents the one-dimensional sequence flattened from the output result of CNN network. $W^{(k)}$ represents the coefficient matrix of the k layer of the network, and $b'^{(k)}$ represents the offset of the k layer of the full connection layer.

3.5. Case Study. This paper uses the random time-series faults to verify the functionality and performance of the proposed model. Figure 5 shows one of the original data used in this study case; this time-series matrix has 91×300 in the dimension.

The study case sets the indexes $W = M = 30$ and $\ell = 1$ to construct the sliding window sequence. 271 sliding time windows will be generated under this parameter selection, and these submatrices will be marked in terms of the relationship between the time series and the fault state. Figure 6 shows the classification process after selecting one of the windows.

After the time-series submatrix with $L = 30$ is intercepted in Step ①, the dimension of the submatrix is reduced. And then, it produces a time-series submatrix with $n = 30$ dimension, with the gray image shown in Step ② of Figure 6. After the feature extraction is performed by CNN, the time-series submatrix becomes a 4×4 submatrix as is shown in the Step ③ in Figure 6.

The output result is an abstract description of the original submatrix. Some original features of the characteristic submatrix can still be shown in the output feature extraction matrix (such as vertical gradient color bands, and horizontal gray difference). After the feature extraction is completed, the number of parameters is reduced from 900 to 256. In this case, the first full connection layer with 256 hidden layers could reduce 164864 hidden layer parameters, which greatly improves the detection performance and achieves high accuracy.

Finally, through the full connection, the probability of the fault detection can be generated, as is shown in Step ④ of Figure 6. It can be seen that the precise fault is the first type of fault, namely, the heat pipe water loss accident, which is consistent with the labeled information.

4. Simulation Experiments

4.1. Experimental Setup

4.1.1. Data set for Experiments. The fault data set of NPP in this study is generated from a simulation software named PCTran/AP1000, as is shown in Figure 7. The software is a reactor transient and accident simulation software, and its reactor model has been widely applied for the NPP system simulation [23].

In this study, ten fault states are set in PCTran/AP1000 software which supply all the train data set and test data set in experiment, the sampling period h is 1 s, and the sample are chosen by 7 different initial conditions, and every condition has a time series with 5 minutes running data. These fault states are merged with the normal state, and the OneHot coding mode is used to code the fault category. The faults and their parameters which can make the model detect faults more sensitively are given in Table 1.

4.1.2. Neural Network Parameter Setting. Since the experiment could describe as series of partial differential equations, the CNN model's convolution layers can be seen as numerical simulation of partial differential equations so that the parameters of the layers do not need too many. Therefore, this study investigates the possible structure of CNN network under this data set and gives a reference structure with the 3 lays and 3×3 convolution kernel shown in Table 2,

4.1.3. Data Preprocessing Settings. This section will discuss the influence of the sliding time window size w, the pivot feature number m, and the selection of the sliding window step size ℓ on the fault detection results. This study will experiment with different parameters based on the network structure in Table 2.

Parameters w and m are the key data for preprocessing the initial samples. On the one hand, their values determine the number of features retained in the initial sample. On the other hand, considering the significance of generating a multiparameter time series that is easy to be processed by the CNN network, setting w is equal to m so that the input matrix is a square matrix.

In addition, this study also tests the step ℓ of the sliding window to prevent over-fitting caused by too dense intervals and under-fitting caused by too sparse intervals.

The performance of the approach to select these two parameters is evaluated in this experiment, which noted as K, and Figures 8 and 9 are the results of the experiments.

Based on the results of Figures 8 and 9, if the K value is too large, the data have more noise and the convexity of the optimization model less significant so that make the model's accuracy has such large fluctuations. If the ℓ is too large, the data set is quite less information so that make the model's accuracy lower.

Specifically, it can be seen from Tables 3 and 4 that after comprehensively considering the indicators such as the average accuracy rate, the best accuracy rate, and variance, the best fit is to set the parameter K to the value 30.

In addition, the difference of ℓ will generate a different number of feature time series, and the accuracy shows a trend that the more images, the higher the accuracy in the data set. Therefore, in this data set, only the relationship between training time and accuracy needs to be considered. Thus, the value of the parameter l is set to 2.

Therefore, the experiment will evaluate the performance of CNN based on the preset parameter values: $K = 30$ and $\ell = 2$.

4.2. Discussions on Experimental Results

4.2.1. Fault Feature analysis. Through PCA dimensionality reduction and sliding window operation, the sliding window time subsequence of the principal component feature of each fault sample data is obtained. Figure 10 is the feature grayscale image of a typical sliding window selected from 11 types of state faults (including 10 types of faults and 1 type of normal state). It can be seen from the result that there are significant differences in the stripes of the grayscale images of different fault types.

The sliding window time subsequence goes through the convolutional layer to further extract fault features and generates a feature submatrix set through convolutional operation. Figure 11 lists the grayscale images of the characteristic submatrix at the time of 24 s, 48 s, 72 s, and 96 s for the normal state, the heat pipe water loss accident, and the Main steam isolation valve closed accident, respectively.

It can be seen from Figure 11 that the feature submatrix sets of the same fault type at different times of the time series have high similarity, as shown in Figures 11(a)–11(i). The average grayscale difference between the grayscale images of each feature submatrix set ranges from the lowest 41.1 to the highest 113.077. However, the feature submatrix grayscale images of different fault types have great differences, and the value range is [188.213, 225.166], as shown in Figures 11(j)–11(l). In addition, it can be observed that the image brightness of Figures 11(j)–11(l) is significantly higher than that of Figures 11(a)–11(i), which fully demonstrates that the sliding window time subsequence can effectively extract the fault features after being processed by the convolution layer.

4.2.2. Diagnostic Accuracy Analysis. In order to verify the accuracy and effectiveness of the model in detecting faults, the generated data sets are used for 20 independent training sessions, and the results are compared with the traditional classic back propagation (BP) algorithm, one part in our own method, LSTM method, and the classic CNN algorithm in [14], which just generate image without the time dimension but simply repeating the single feature vector. The comparison indicators include the accuracy and different training variances, and the results are shown in Figure 12.

The experimental results show that the accuracy and training variances of the CNN models based on time-series method and the sliding window mechanism are significantly better than those of the BP neural network. In terms of accuracy, the average accuracy of the BP neural network is only 67%, the average accuracy of the LSTM method is 86.9%, while the average accuracy of the model proposed in this

Table 1: Fault category, coding, and parameters.

Condition type	Numbers	Label	Fault setting in PCTran
1. Normal state	2107	10000000000	−0 0 0 0 0 0 0 0 0
2. Heat pipe water loss accident	2107	01000000000	1% of 100cm^2 1 0 0 0 0 0 0 0 0 0
3. Cold pipe water loss accident	2107	00100000000	1% of 100cm^2 0 1 0 0 0 0 0 0 0 0
4. Steam pipeline inside containment	2107	00010000000	1% of 100cm^2 0 0 1 0 0 0 0 0 0
5. Steam pipeline outside containment	2107	00001000000	5% of 100cm^2 0 0 0 1 0 0 0 0 0

Table 1: Continued.

Condition type	Numbers	Label	Fault setting in PCTran
6. Water supply accident	2107	00000100000	-0 0 0 0 1 0 0 0 0 0
7. Main steam isolation valve closed	2107	00000010000	-0 0 0 0 0 1 0 0 0 0
8. Coolant pump rotor stuck	2107	00000001000	-0 0 0 0 0 0 1 0 0 0
9. Turbine accident shutdown	2107	00000000100	-0 0 0 0 0 0 0 1 0 0
10. Steam pipe A pipe broken	2107	00000000010	1% of 100cm^2 0 0 0 0 0 0 0 0 1 0
11. Steam pipe B pipe broken	2107	00000000001	1% of 100cm^2 0 0 0 0 0 0 0 0 0 1

Table 2: CNN network parameters.

Structure name	Network structure
Convolution layer 1	Number of filters:8 Kernel size: 3 × 3 Activation function:ReLu
Max-pooling layer 1	Pool size: 2 × 2
Convolution layer 2	Number of filters:16 Kernel size: 3 × 3 Activation function:ReLu
Max-pooling layer 2	Pool size: 2 × 2
Convolution layer 3	Number of filters:16 Kernel size: 3 × 3 Activation function:ReLu
Max-pooling layer 3	Pool size: 2 × 2
Fully-connected layer	Nodes: 32; activation function: ReLu
Output layer	Nodes:14; activation function: Softmax
Optimization algorithm	Adam algorithm (learning rate: 0.001; beta1: 0.9; beta2: 0.99)
Loss function	Sparse_categorical_crossentropy

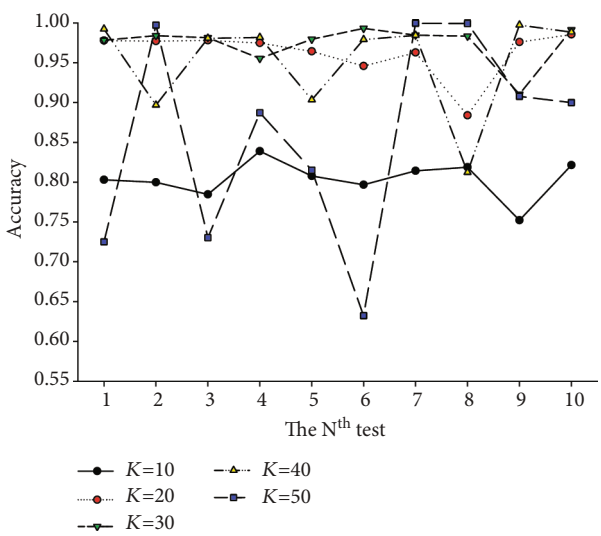


Figure 8: Accuracy of different K values when ℓ is equal to 2.

paper is increased to 99%. In addition, the optimal accuracy of the BP neural network is 98.7%, and the optimal accuracy of the LSTM method is 90.9%, while the proposed model of this paper is improved to 99.8%. In terms of variance, the variance of BP neural network is 0.091, and the variance of LSTM is 0.0027, while the model proposed in this paper is only 0.0007.

For the comparison of the classification effect, this paper selects the confusion matrix of the four methods' one of the typical experiments for comparison. The results are shown in Figure 13. It can be seen that in the classification of the normal state, the CT-CNN method has a very good performance that only 1% of the normal state data is misdiagnosed; this conclusion is better than other methods. The difference between the normal state and the fifth fault, steam

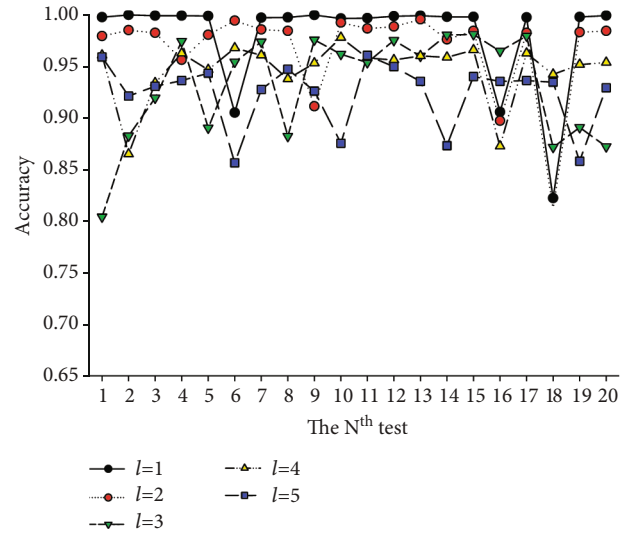


Figure 9: Accuracy of different ℓ values when K is equal to 30.

Table 3: Accuracy of the different K values when l is equal to 2.

K value	10	20	30	40	50
Average accuracy	0.804	0.967	0.961	0.951	0.859
Best accuracy	0.839	0.987	0.994	0.997	0.999

Table 4: Accuracy of different l values when K is equal to 30.

ℓ value	1	2	3	4	5
Average accuracy	0.979	0.961	0.932	0.947	0.923
Best accuracy	0.999	0.994	0.980	0.977	0.960

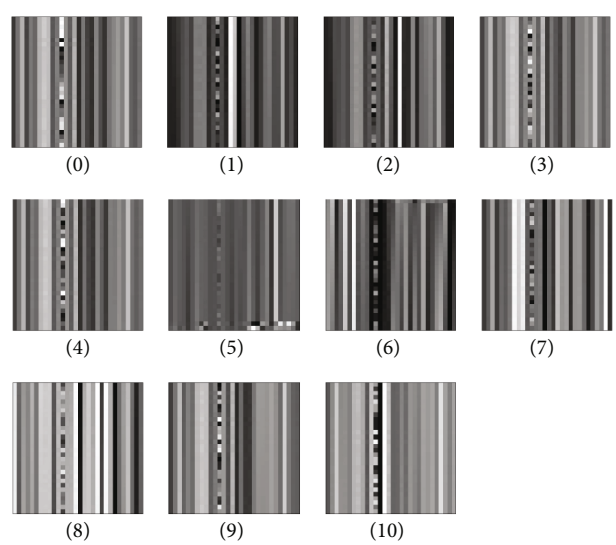


Figure 10: Subsequence grayscale image of sliding window for 11 types of faults (including normal state).

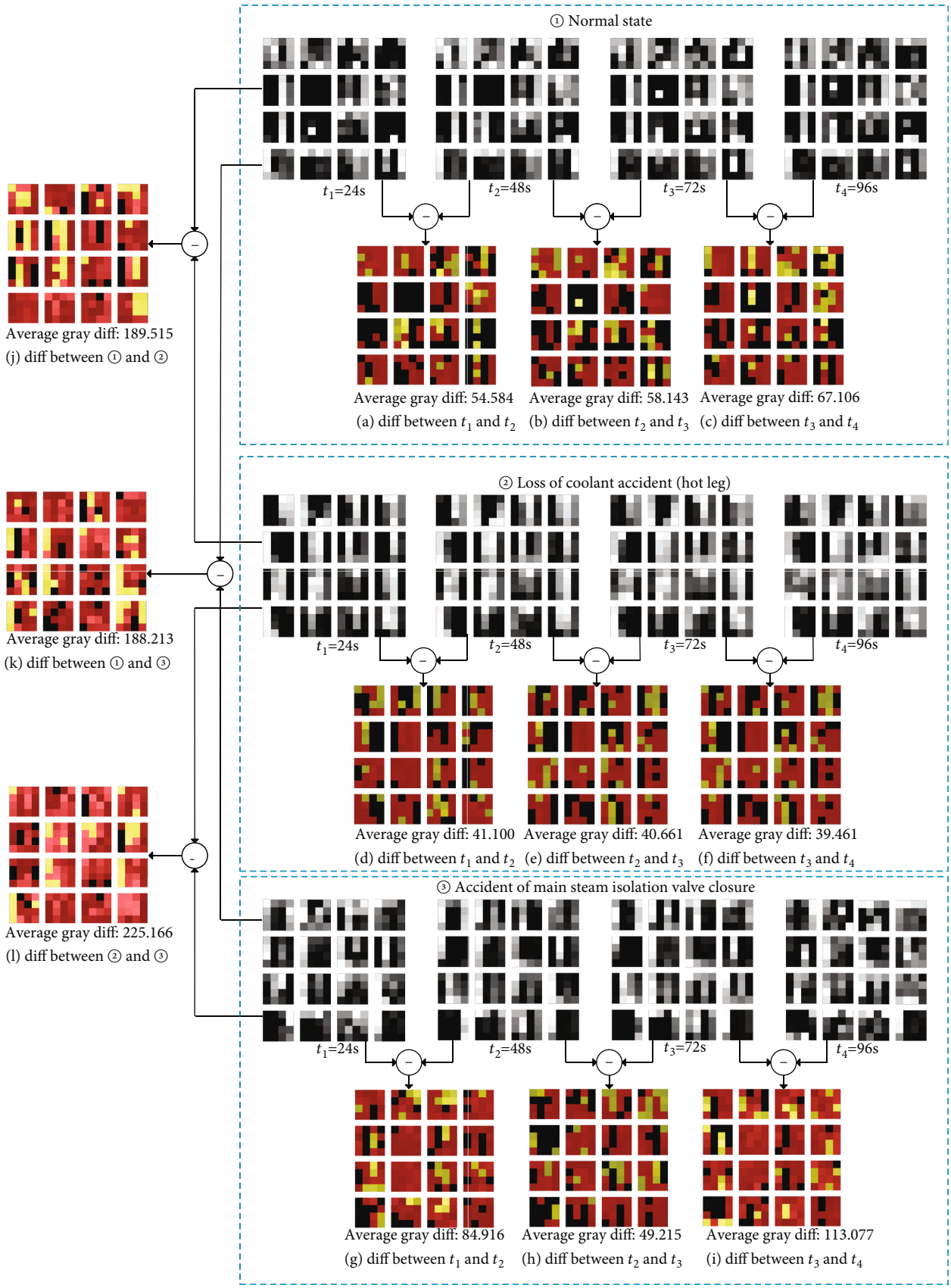


Figure 11: Grayscale comparison of different faults at different times.

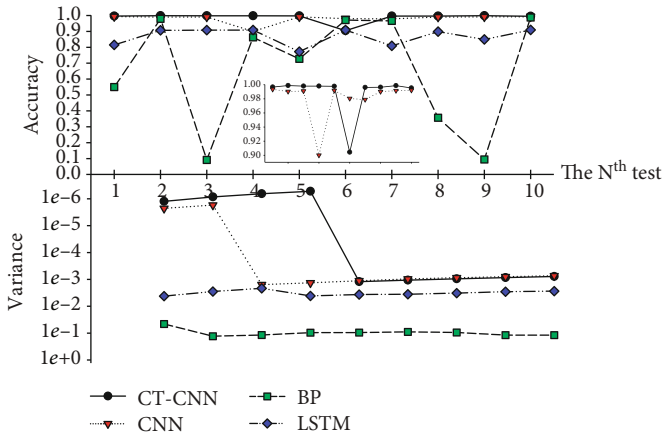


Figure 12: Accuracy comparison with four methods.

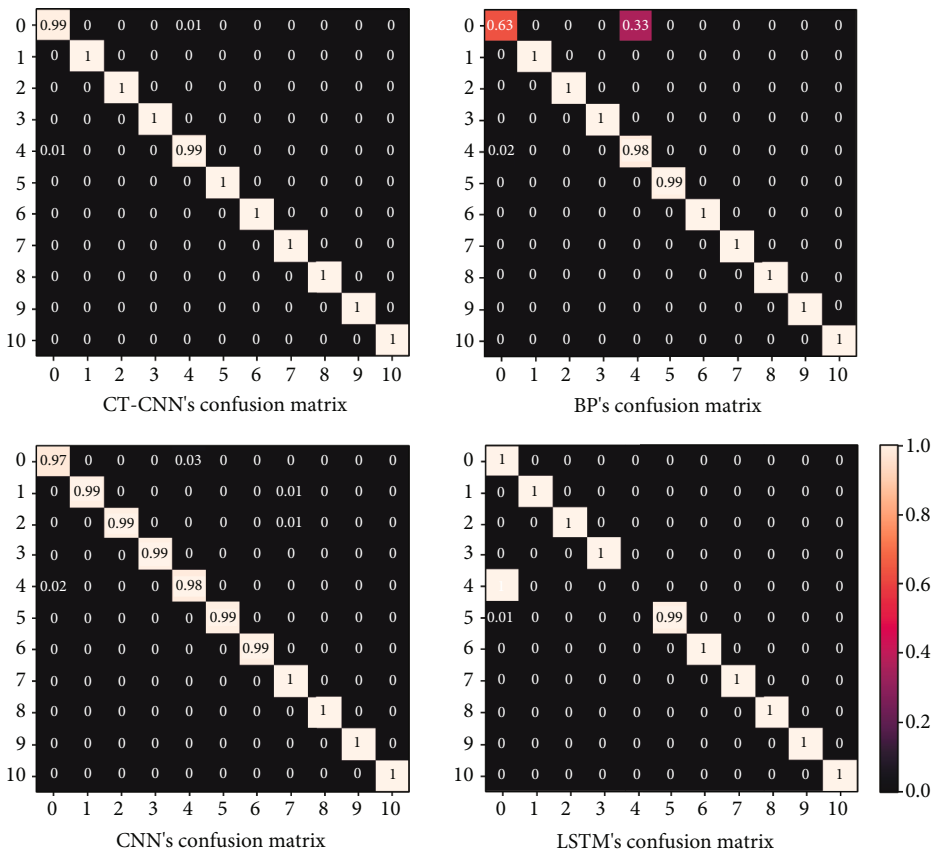


Figure 13: Confusion matrix comparison with four methods.

pipeline outside containment, can be divided, and all faults can be predicted more accurately.

Compared with the standard CNN method, the model proposed in this paper also shows the improvement in accuracy. The highest accuracy rate of the standard CNN method is 99.3%, and the average accuracy rate is 98.0%. By using the approach in this paper, the highest accuracy rate is improved by 0.5%, and the average accuracy is increased by 0.8%. In addition, the variance of the two methods is almost the same.

The above results show that the model proposed in this paper has strong stability and accuracy.

5. Conclusion

This paper proposes a FD mechanism for NPP control system, which is based on PCA, time-series analysis, and CNNs. The CNNs process the time-series data collected by the NPP system through the time window, which not only retains the organic features of the internal time and state information of

the data, but also reduces the processing difficulty and improves the feasibility of the fault data. In addition, the proposed approach can be adaptive to different data set and demonstrates a stable training process. The experimental results show that the proposed approach can achieve better performance in both detection accuracy and variance than the classic back propagation (BP), LSTM, and standard CNN algorithms. More significantly, the optimal accuracy of the proposed model can be as high as 99.8%.

Data Availability

All data included in this study are available upon request by contact with the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the Science and Technology on Thermal Energy and Power Laboratory Open Foundation of China (No. TPL2019C01).

References

[1] M. El-Sefy, A. Yosri, W. El-Dakhakhni, S. Nagasaki, and L. Wiebe, "Artificial neural network for predicting nuclear power plant dynamic behaviors," *Nuclear Engineering and Technology*, vol. 53, no. 10, pp. 3275–3285, 2021.

[2] Y. Wu, "Development and application of virtual nuclear power plant in digital society environment," *International Journal of Energy Research*, vol. 43, no. 4, pp. 1521–1533, 2019.

[3] M. Peng, H. Wang, X. Yang et al., "Real-time simulations to enhance distributed on-line monitoring and fault detection in pressurized water reactors," *Annals of Nuclear Energy*, vol. 109, pp. 557–573, 2017.

[4] L. Breiman, "Random forests[J]," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] S. Bhardwaj, S. Tewari, and S. Jain, "Study on future of artificial intelligence in neural network system," *International Journal of Scientific & Engineering Research*, vol. 4, no. 6, pp. 597–601, 2013.

[6] J. Li and M. Lin, "Ensemble learning with diversified base models for fault diagnosis in nuclear power plants," *Annals of Nuclear Energy*, vol. 158, article 108265, 2021.

[7] H. Wang, M. J. Peng, J. W. Hines, G. Y. Zheng, Y. K. Liu, and B. R. Upadhyaya, "A hybrid fault diagnosis methodology with support vector machine and improved particle swarm optimization for nuclear power plants," *ISA Transactions*, vol. 95, pp. 358–371, 2019.

[8] Y. Yu, M. Peng, H. Wang, Z. G. Ma, and W. Li, "Improved PCA model for multiple fault detection, isolation and reconstruction of sensors in nuclear power plant," *Annals of Nuclear Energy*, vol. 148, article 107662, 2020.

[9] L. Yong-kuo, P. Min-jun, X. Chun-li, and D. Ya-xin, "Research and design of distributed fault diagnosis system in nuclear power plant," *Progress in Nuclear Energy*, vol. 68, pp. 97–110, 2013.

[10] Y. Ren, D. Fan, X. Ma, Z. Wang, Q. Feng, and D. Yang, "A GO-FLOW and dynamic Bayesian network combination approach for reliability evaluation with uncertainty: a case study on a nuclear power Plant," *Ieee Access*, vol. 6, pp. 7177–7189, 2018.

[11] J. Choi and S. J. Lee, "Consistency index-based sensor fault detection system for nuclear power plant emergency situations using an LSTM network," *Sensors*, vol. 20, no. 6, p. 1651, 2020.

[12] Y. Yao, J. Wang, M. Xie, L. Hu, and J. Wang, "A new approach for fault diagnosis with full-scope simulator based on state information imaging in nuclear power plant," *Annals of Nuclear Energy*, vol. 141, article 107274, 2020.

[13] B. S. Peng, H. Xia, Y. K. Liu, B. Yang, D. Guo, and S. M. Zhu, "Research on intelligent fault diagnosis method for nuclear power plant based on correlation analysis and deep belief network," *Progress in Nuclear Energy*, vol. 108, pp. 419–427, 2018.

[14] J. Li and M. Lin, "Research on robustness of five typical data-driven fault diagnosis models for nuclear power plants," *Annals of Nuclear Energy*, vol. 165, article 108639, 2022.

[15] M. Xia, T. Li, L. Xu, L. Liu, and C. W. de Silva, "Fault diagnosis for rotating machinery using multiple sensors and convolutional neural networks," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 101–110, 2018.

[16] Y. Yao, H. Wang, S. Li et al., "End-to-end convolutional neural network model for gear fault diagnosis based on sound signals," *Applied Sciences*, vol. 8, no. 9, p. 1584, 2018.

[17] C. C. Chen, Z. Liu, G. Yang, C. C. Wu, and Q. Ye, "An improved fault diagnosis using 1D-convolutional neural network model," *Electronics*, vol. 10, no. 1, p. 59, 2021.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[19] Y. Li, Z. Hao, and H. Lei, "Survey of convolutional neural network," *Journal of Computer Applications*, vol. 36, no. 9, p. 2508, 2016.

[20] C. He, D. Ge, M. Yang, N. Yong, J. Wang, and J. Yu, "A data-driven adaptive fault diagnosis methodology for nuclear power systems based on NSGAII-CNN," *Annals of Nuclear Energy*, vol. 159, article 108326, 2021.

[21] Z. Wang and T. Oates, "Imaging time-series to improve classification and imputation," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, Buenos Aires, Argentina, 2015.

[22] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[23] M.-S. Technology, *PCTRAN/U3LP personal computer analyzer of PWR 3-loop, version 4.2.1, 2009*, 2009.