*Journal of Imaging Science and Technology*® 66(4): 040406-1–040406-13, 2022.


# DiM-PCNet:3D Point Clouds Classification with Multi-scale and Multi-level Feature Net

**Kun Zhang, Liting Zhang, Xiaohong Wang, Xinshuai Hua, and Xuan Gao**
*Department of Information Science and Engineering, Hebei University of Science and Technology, Shijiazhuang, Hebei 050018, China*
*E-mail: zhangkun@hebust.edu.cn*

**Abstract.** *Nowadays, point clouds are frequently gathered by 3D scanners such as Lidar and Kinect, which produces thousands of point cloud models. Point cloud processing is vital to 3D vision, especially 3D object recognition, positioning, and navigation technology. Addressing uneven data density caused by coordinate frame transformations and the inherent problem of insufficient context connection in point clouds, the DiM-PCNet (Multi-scale and Multi-level Point Clouds Classification Network, and the Di is a prefix to represent double M in Multi-scale and Multi-level) is proposed in this paper. DiM-PCNet is provided for object classification with multi-scale and multi-level features. We encode the point cloud in multi-scale and fully fuse the features with the raw point cloud for keeping the context relationship. In DiM-PCNet, we sample the point clouds from eight parts for multi-scales feature extraction. The multi-scales features are fully fused by multi-level pyramid models. The multi-scale and multi-level strategies are applied in DiM-PCNet, in which the abundant and important features of point clouds are extracted and utilized in the 3D object classification. It is worth noting that the DiM-PCNet feature block can be embedded into the segmentation net, where the accuracy achieved is 87.1%. We conducted experiments on ShapeNet and ModelNet40 and the experimental results show that DiM-PCNet achieves state-of-the-art performance in 3D object classification. The experiment shows competitive performance on robustness and segmentation tasks.*

[DOI: 10.2352/J.ImagingSci.Technol.2022.66.4.040406]

## 1. INTRODUCTION

In general, 2D images provide limited information that cannot satisfy applications such as autonomous driving [1], robotics [2], and mixed reality [3]. Recently, more and more research studies focus on 3D vision so that abundant depth and geometric data for measured objects emerge. 3D object classification also plays an important role in large amounts of 3D vision tasks. For example, different objects have to be classified by detailed features like the shape or outline of the objects, which further produces suitable actions in autonomous driving.

Recently most of the point cloud classification applies the strategy of a single-scale. However, the single-scale has limited robustness in the classification network. If the uneven point cloud data are input into single-scale network, classification accuracy would reduce. Compared with the multi-scale network, the single scale network only gets the feature of one scale, but the multi-scale network gets two, three or more. For classification and segmentation of point cloud, mode features will reduce the impact of missing parts on the whole. The point cloud classification with multi-scale captures diverse and complete data features, which reduces the impact of point cloud data imbalance on point cloud classification [4]. In order to retain the point cloud's details, we increase the number of levels of the multi-scale network. The level is the convolution layers and size, which is based on the pyramid model. We are inspired by SPP, however, it is actually different from the pyramid model we mentioned. SPP convolutes first and then fuses the pyramids model to pooling, but our network uses the pyramid model in convolution.

In addition, to avoid slow response in the classification net, downsampling is acted on point cloud data sets.

In this paper, we mainly consider how to identify the unbalanced or sparse point cloud data with higher accuracy and efficiency in a deep learning network. For multi-scale feature extraction of the point cloud, the significant features are enhanced by the aggregation of different scales, which get the different foci.

The baseline of DiM-PCNet is PointNet, and the features are also extracted by MLP (Multi-Layer Perceptron). However, the features of DiM-PCNet are propagated by the pyramid model. The convolution levels of ours are more than PointNet. Another key point is the stability and adaptability of the data. We have downsampled the data set and will not burden the computer. In this paper, we design mechanisms of multi-scale and multi-level into a point cloud processing architecture. Our contribution can be summarized as follows:

- For balancing distribution of point cloud data, the direction consistency sampling algorithm is designed along eight quadrants shown in Figure 1. After that, a point cloud classification network with multi-scale and multi-level was built to extract the detailed features. In the classification network, the multi-scale and multi-level strategy is applied for improving the accuracy of the point cloud classification network.
- The detail features are extracted by DiM-PCNet, and form the pyramid layer [5] as each scale. Especially, the gathered data by Lidar is inevitably sparse and uneven. The pyramid layer features are stable and robust for the classification result.

Received Nov. 10, 2021; accepted for publication Mar. 16, 2022; published online May 11, 2022. Associate Editor: Ping Wang.
1062-3701/2022/66(4)/040406/13/$25.00

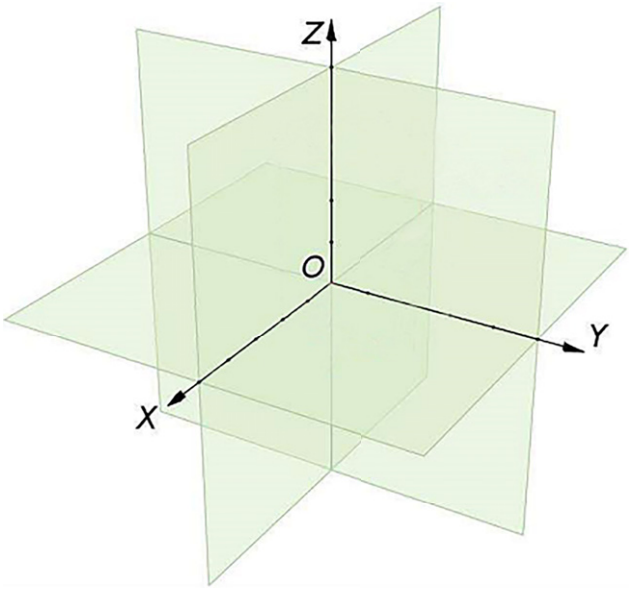



Figure 1. Three dimensional quadrant coordinate diagram.

- We verify the classification net proposed in this paper which improves the results on the ModelNet40 and ShapeNet by experiment. Especially the accuracy rate of ShapeNet is 97.9%, and the number of categories with an accuracy of 100% in ModelNet40 is increased.

## 2. RELATED WORK

In studies of 3D data [6–8], we found that point clouds were provided as the visualized expression for the 3D object, and the point clouds have simple and limited data attributes which describe the 3D scene mainly as independent three dimension coordinates. Unlike voxel meshes and patches, the point clouds could be represented by scatter data without topology structure. Nevertheless, the expression of point cloud form is flexible, and it has outstanding advantages in the expression of shape and geometric characteristics. The point cloud can be directly acquired by 3D sensing equipment such as Lidar, and the data is highly reliable. 3D vision has greater development prospects than 2D vision, and it also has more difficult challenges than 2D vision. The data of the 2D image is relatively stable, with pixel value and low-level texture information, but the 3D data structure is relatively unstable. For example, the 3D data of the point cloud has disorder [9] and rotation [10], and it's difficult for the convolution of point cloud .

There are several methods for the convolution of the point cloud.

### 2.1 *Convolution based on 2D Image Transformed from Point Cloud*

The first method is to convert the point cloud into a few 2D images. The fact is that inputting the 2D image to the convolution network to get the results of classification. Based on the 2D image, the multi-view [11–13] is developed to represent the point cloud, and the classic and mature 2D deep neural network is used for learning and training, which has achieved good results in the recognition of 3D objects. However, due to the need to collect images from multiple perspectives, there are problems such as data redundancy and angle selection, and the 3D structure information of the object is not really used. At present, most researchers prefer to use 2D image information as an auxiliary to 3D data, and the information of 2D applied to multimodal recognition [14].

### 2.2 *Convolution based on Voxel Grid Transformed from Point Cloud*

The second method is to convert the point cloud into the form of a voxel grid, each grid has a value and each grid has a fixed size, and a convolution operation is performed on this basis. Some researchers [15–17] transform the 3D point cloud into voxel mesh and input it into the neural network for deep learning. For example, ModelNet40 set up the voxelized versions and input them into VoxNet [18] for object classification. However, there are some problems in the method of point cloud voxelization, such as voxels lacking color information, increase computational complexity with the difference in voxel's resolution, and only the global features of voxels being considered in the classification. So researchers began to focus on the convolution of point cloud directly.

### 2.3 *Convolution based on Point Cloud*

The third method is to use the MLP operation of shared weights to perform point-by-point convolution operations after PointNet. The PointNet [19] was proposed by the Charles' team in 2016, which solves the problem of the disorder and rotation of point clouds by symmetric functions and T-Net. The PointNet is the beginning of a milestone in which point clouds are directly inputted for convolution. After PointNet, increasing amount of research has focused on using deep learning directly on the point cloud instead of voxel grids. However, the PointNet does not make full use of the feature of local points [20] in the point cloud. In order to solve this problem, Charles' team proposed PointNet++ [21] in 2017, which groups the points after farthest point sampling to extract features.

Because PointNet processes point clouds by computing individual points, it does not consider the information of local point clouds and lacks the use of fine-grained point cloud information. Later, PointNet++ divided the point cloud into multiple neighborhoods and grouped them for multi-layer perceptron operation. In addition, PointNet++ adds a multi-scale module to reduce the impact of uneven density point clouds on the overall recognition effect, which has also been verified that considering multi-scale information is more robust than PointNet. So a multi-scale strategy is better than a single-scale strategy. Our work belongs to the third method, and after obtaining feature abstraction between points, we use multi-scale and multi-level features to make full use of the features of each part of the point cloud. However, PointNet ++ uses KNN or ball query to determine

the range of scale. It is not as good as our network directly, according to the number of points and eight quadrants to determine the scale range. Both KNN and ball query have higher time complexity than ours. Both DGCNN [22] and SpiderCNN [23] also use KNN. Although DCG-Net [24] improves the convolution method that related approaches ECC [25] by dynamic routing mechanism, the computer is required to have high computing performance.

### 2.4 *Point Clouds Classification based on Extracting Features of Multi-scale*

Specifically for the uneven point cloud, PointNet++ imports the MSG (multi-scale grouping) and MRG (multi-resolution grouping) and shows that the accuracy, as considering sampling and multi-scale features, is improved. Then the performance of the deep learning based on multi-scale features is increased gradually. Chen [26] improves the MVSNet and builds a deep learning network (Point-MVSNet), which is to fuse the features with a multi-scale of 2D and 3D. Engelmann et al. [27] divide four blocks to enhance the local connection between points and score the point cloud classification merging the multi-scale features. Xiaodong Zhang et al. [28] use a multi-scale region proposal network and a multi-scale object detection network to reality object detection of remote sensing imagery, and propose the ACNMS strategies to promote detection performance. Zhongyang Zhao et al. [29] build a classification network of the scale point cloud, which is determined by the neighborhood sphere, for Lidar point clouds of ground buildings. Bai Jing et al. [30] propose a multi-scale point cloud classification network (MSP-Net), which joins a local area division method to control the size of the local scale during feature extraction. Liu Yongcheng et al. [31] propose a point cloud DensePoint network, which repeatedly aggregates multi-level and multi-scale features to enhance robustness. However, they do not consider the density of the local point cloud, so the accuracy decreases when the point clouds are unevenly distributed. We now know that point cloud sampling [32] reduces the influence of the uneven distribution of point clouds on the accuracy from [20].

### 2.5 *Framework of Dim-PCNET*

Point cloud classification with deep learning is a data-driven processing method that has high computing performance in deep feature extraction and participates effectively in classification prediction. Assumed $n$ is the size of the point clouds, which input $n$ points into the network. In the sampling layer, we denote the DiM-PCNet input of scales as $s$ ($s = 1, 2, \ldots, m | m \in N$). Figure 2 is a diagram of the time with different scales.

In the DiM-PCNet, the point cloud is set as $P$, where $P_i$ represents the $i$th point in $P$, which is computed by three-dimensional orthogonal coordinate $x, y, z$ in Figure 3. of raw point clouds. The architecture of the DiM-PCNet is shown in Fig. 3, which consists of four layers: sampling layer, convolution layer, aggregation layer, and full connection layer. We divide the point clouds into different scales. Each scale represents different sparsity of point clouds.

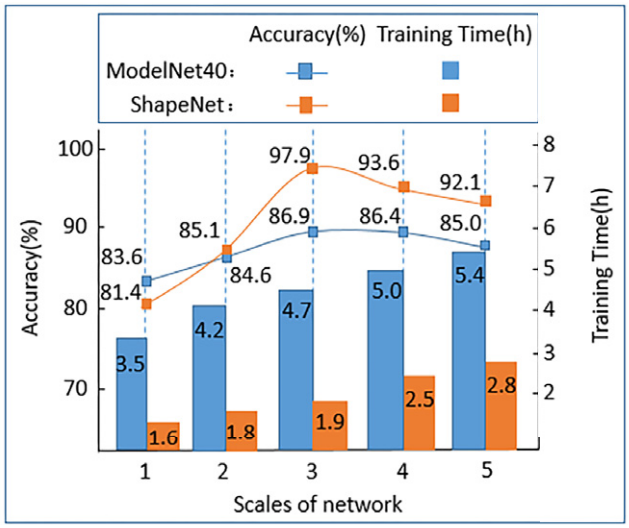



Figure 2. Diagram of the accuracy and time with different scales.

Note that for the complexity of computation and the limitation of time, we only select $s = 3$. But the framework is not limited to 3 scales-3 levels. The different scales are generated by point cloud sampling. The boundary box of the point cloud can be calculated, which is divided into equivalent 8 parts by point cloud sampling, random sampling algorithm is used in each boundary box.

In our sampling algorithm, we select 8 parts to sample that the point cloud describes the object in a 3D space, and it is balanced for 8 quadrants division in the three-dimensional coordinate. If the 2 parts or 4 parts are divided in the net, the uniformity of the point cloud with random sampling in each part is unsatisfied.

In the convolution layer, the sampling results are used as the input of the convolution layer. For the different scales, we try to extract the scale's features by a similar method. The T-Net is joined in the datasets after sampling to maintain the geometric invariance of the point cloud. The first level, second level, and third level of the pyramid input are T1, $T1 + T2$, and $T1 + T2 + T3$ respectively. After that, T1, T2, and T3 are combined to obtain feature T4 with different dimension features. T4 is a multi-scale strategy for point cloud features. The max-pooling is used on T4 to get the feature value ST that is the output of the DiM-PC block. In the aggregation layer, SP is the superposition of three different scales features (ST). The superposition rule is to keep the vector size of the output the same as the input. In a nutshell, we convert the coordinate points of the data set into tensors, and the dimension of tensors is increased by MLP that the network struct is a pyramid model. After max-pooling, the tensors become vectors, which is the features extraction. Three scales produce three vectors that have different sizes. Finally, the addition of vectors is the aggregation of the features.

In the full connection layer, the features are selected step by step. The output dimension of the aggregation layer is reduced from 5376 to 1024, and then to 512. The dimension

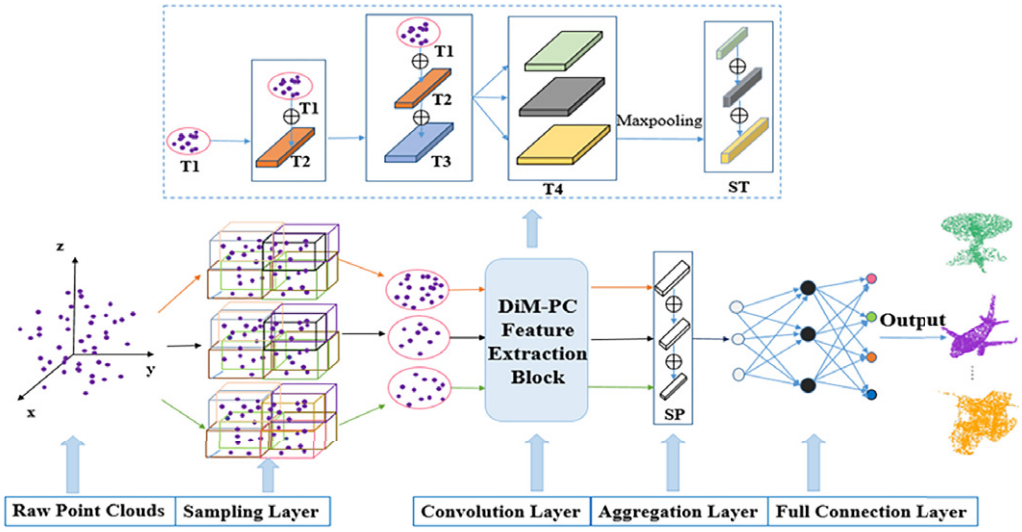



Figure 3. The overall network architecture of DiM-PCNet.

reaches 256 from 512 with dropout. The output of the full connection layer is K that represents the number of 3D objects, which is the prediction value of the classification network model. After learning and training, accuracy is gradually improved until to achieve a satisfied classification effect.

## 3. METHOD

### 3.1 *Direction Consistency Sampling*

For most multi-scale networks, the time complexity is greater than the single-scale networks. So we try to adopt a method with small time complexity in the part of the point cloud sampling. By comparing the time complexity of Farthest Point Sampling (FPS) [33], Inverse Density Importance Sampling (IDIS) [34], and Random Point Sampling (RPS), the time complexity of RPS is lower and can save the running time. However, the sampling uniformity of RPS is slightly worse. In the DiM-PCNet of sampling, a method of evenly dividing the box before RPS in this paper. Previously, we calculated that the density of the data set is uniform, which is guaranteed uniformity. To pursue efficiency, the random sampling algorithm is applied along with one direction sampling of the point cloud. Although the precision of point clouds is slightly lower after the random sampling algorithm, [34] has proved the random sampling significantly reduces the burden of the network and saves the running time of DiM-PCNet. There are two intentions to divide boxes. One is to disperse the point cloud so that it will not be unevenly collected during random sampling. The other is to form the different scales of point clouds. The calculation equation of the boundary box is shown in Eq. (1).

$$L_d = \left|\mathrm{Max}\left(P_i^d\right) - \mathrm{Min}\left(P_i^d\right)\right|, P_i^d \in P, \tag{1}$$

where $L_d$ is the length of the boundary box, and $d$ represents the three orthogonal directions ($x$, $y$, and $z$) in the coordinate system.

The maximum distance between points in the $x$, $y$, and $z$ directions ($L_x$, $L_y$, and $L_z$) can be obtained by Eq. (1). Then the center point (defined as $P_c$) of the boundary box is used as the origin in a new coordinate system that has 8 quadrants. So the box is divided into 8 parts on average.

$$\eta = |P_{Br}|, r = 1, 2, 3, \ldots, 8, \tag{2}$$

The $\eta$ is the number of elements in the set of $1, 2, \ldots, r$ boxes. The $P_{Br}$ is the point cloud in the $1, 2, \ldots, r$ boxes. The $B$ represents the box, and $r$ is the number of the boxes that you need to choose in Eq. (2).

$$\mathrm{PRS} = f^{RS}(\varphi(\eta), P_{Br}), r = 1, 2, 3, \ldots, 8, \tag{3}$$

The $PRS$ is the point cloud of sampling in the $1, 2, \ldots, r$ boxes. If the $r$ is 8, the $PRS$ is the whole point cloud data set of sampling. In Eq. (3), $\varphi$ is a function that points of boxes match with batch-size. $f^{RS}$ represents a random sampling of each box.

For DiM-PCNet, the direction consistency sample is a key step in which the local sampling algorithm is implemented in point cloud along with eight quadrants respectively. The eight direction offers the average contribution to DiM-PCNet, which guarantee point cloud uniformity.

### 3.2 *Extracting Feature in DiM-PCNET*

The DiM-PC feature extraction block is introduced in DiM-PCNet, and retains more point cloud's details, which increase the number of levels of DiM-PCNet, then it is possible for revealing the local feature. If the feature extracted repeatedly is a wing, it means that the wing has a greater impact in the classification.

The high dimensional features of the pyramid form the superposition layer by layer. Feature extraction is shown in

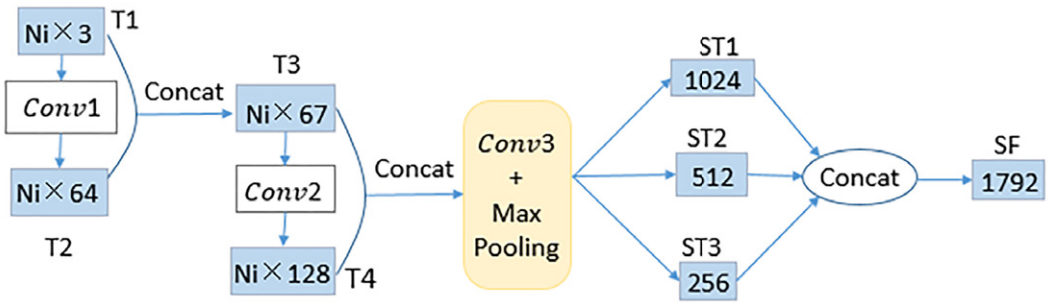



Figure 4. Schematic diagram of multi-level feature extraction process.

Figure 4, and the feature extraction can be formulated as

$$F = \sum_{s=1}^{S} \sum_{j=1}^{J} \phi \left( Conv_j \left( P_i \right), s \right), \forall P_i \in P, \tag{4}$$

where $F$ represents the output of the extracted feature, and the $s$ represents the scale of sampling points. The function of $\phi$ is a fuser of $S$ scales, and $Conv_j(p_i)$ represents the convolution operations which contain $j$ channels on the features of point cloud. Each scale has the same convolution operations and the output of each convolution is a deeper feature. Point clouds of different scales are obtained from the sampling in Eq. (3), such as scale1, scale2, and scale3 in Fig. 3.

As we can see the Fig. 4, the input of the point cloud is recorded as T1 whose size is $Ni \times 3$, where $Ni$ is the point number of $i$th scale. The feature T2 of $Ni \times 64$ is obtained by T1 with Conv1. Then we input T3 into Conv2 to get T4, where T3 is a combination of T1 and T2. It is joined low and high features to catch the context relation of the point cloud. T4 is combined with T3 to input Conv3 for three different levels of features. And then max-pooling is used to get three different length feature vectors ST1, ST2, and ST3. Then, ST1, ST2, and ST3 are spliced to obtain SF which is the single-scale feature.

If a more accurate point cloud classification result is required in a certain experiment, the number of scales could be appropriately increased and associated with raising the levels of feature extraction. If there is a large volume of point cloud and a relatively high requirement for net computational efficiency, carefully reducing the network structure is very important. It is worth noting that the multi-scale and multi-level feature extraction part can be used as a feature extraction tool independently, which can be embedded into any other point cloud classification network conveniently.

### 3.3 *Semantic Segmentation*

*(1) Semantic Segmentation Architectures*

The semantic segmentation model takes (B) $\times n \times 6$ as the input, the B is the batch-size of the point cloud. For convenience, B is expressed as 1. The $n$ is the size of the point cloud datasets. 6 represents the number of attributes of the point cloud, which is 3D objects in the scene. The coordinate value $(x, y, z)$ and its color value $(R, G, B)$ can be judged according to the different color values in different 3D objects. The input point cloud data is sampled by the direction-consistent sampling method to obtain the multi-scale scene. The different scale point cloud is shown in the three circled areas in Figure 5. Then, we use the spatial shift upsampling method to enhance the contour information of the point cloud. The local and global features are both produced by the DiM-PC Feature Extraction Block, which obtains more detailed and rich features for segmentation the 3D objects. The global feature is obtained by the max-pooling of features with a size of $n \times 1792$. The multi-scale and multi-level feature extraction module produces two local features. The sizes of them are $n \times 64$ and $n \times 128$, which represents features with different levels of abstraction. Fusing the local features and the global features obtains the features with the size of $n \times 1984$. Finally, the fused features are input into the fully connected network to reduce the dimensionality of the features. The predicted value of each type of 3D object $n \times m$ is obtained, where $m$ is the number of types of 3D objects in the scene.

The feature extraction equation of the segmentation network is shown in Eq. (5).

$$\boldsymbol{F}_{ss} = [F; Conv_2 (P_i)], \tag{5}$$

$F_{ss}$ represents the scene segmentation feature after fusing the global features and the local features. $\boldsymbol{F}$ is a matrix and the output of multi-scale and multi-level point cloud features in Eq. (4). $Conv_2(P_i)$ is also a matrix that is derived from the 2 convolution operations after the DiM-PC feature extraction block. The $[F; Conv_2(P_i)]$ is the vertical concatenation of matrices.

*(2) Spatial Shift Upsampling*

The point cloud spatial shift upsampling is used to increase the number of the points in the point cloud, which strengthen the contour and shape of the point cloud object. The point cloud after directional consistent sampling is divided into different scale point cloud. Next, the point cloud would be upsampled in the different scale point cloud, and then the point cloud is input into the feature extraction block. We reduce the running time and memory by downsampling, and use upsampling to highlight the contour of 3D objects, which

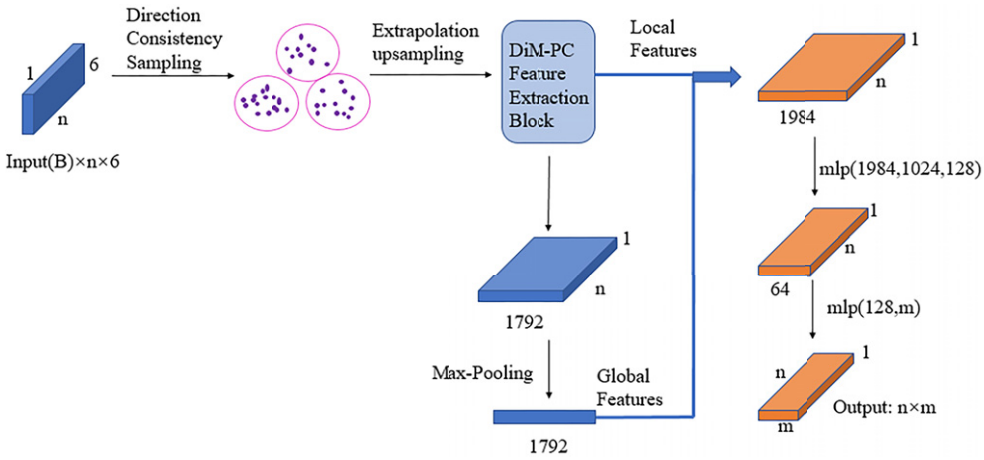



Figure 5. The architecture of Semantic segmentation network embedded with DiM-PC feature extraction block.

is better for the segmentation of different objects in the 3D scene.

The spatial shift upsampling is based on the direction consistency sampling, so it also divided into eight blocks of the same size for upsampling.

The spatial shift in each sample block is shown in the following equation.

$$PUS = P_i^{Br} + \Delta O, \forall P_i \in P, \tag{6}$$

where $PUS$ is the point cloud after spatial shift, $P_i^{Br}$ is the points in the $r$th box, and $\Delta O$ represents the offset vector of points.

The spatial shift upsampling rule is that the point cloud of the odd-numbered block moves to the left, and the point cloud of the even-numbered block is moved to the right. The overall trend is to expand outward, which is to strengthen the edge contour information of the point cloud. The equation of spatial shift rule is shown in Eq. (7).

$$\Delta O = \begin{cases} p^{Br}_{(x-1,y-1,z-1)}(r = 1, 3, 5, 7) \\ p^{Br}_{(x+1,y+1,z+1)}(r = 2, 4, 6, 8), \end{cases} \tag{7}$$

where $p^{Br}_{(x-1,y-1,z-1)}(r = 1, 3, 5, 7)$ represents that the coordinates of points minus one in the first, third, fifth, and seventh box, and $p^{Br}_{(x+1,y+1,z+1)}(r = 2, 4, 6, 8)$ represents that the coordinates of points plus one in the second, fourth, sixth, and eighth box.

If the moved point happens to be an existing sampling point, the moved point required to be corrected:

$$\Delta O_{\text{new}} = \Delta O \pm \Delta O \cdot (\Delta O)^T \cdot P^{Lr}_{(x,y,z+1)}, \tag{8}$$

where $\Delta O_{\text{new}}$ is the new spatial shift rule, and $\Delta O \cdot (\Delta O)^T$ is the module of offset vector. The $P^{Lr}_{(x,y,z+1)}$ is a direction vector which is parallel with $z$ axis. Equation (8) use addition by default, and use subtracts if points still overlap or beyond bounding box.

A schematic diagram of spatial shift, that moves the points of point cloud is shown in Figure 6.

There is a small arrow pointing from one point to another in the II area, which means that the moved point coincides with an existing point. In this case, the moved point will be corrected to ensure that all moved points are newly created independently.

The small arrow of the III area on the right indicates that the point before the spatial shift upsampling is exactly at the boundary, and the point after the spatial shift upsampling is outside the bounding box. The correction of the moved points ensure that all points by spatial shift upsampling are within the bounding box of the original point cloud.

Figure 7 is a visualization of a table. In the figure, the outer contour of the table on the right is obviously thicker than the one on the left.

### 3.4 *Loss Function*

We use the cross-entropy loss function $L_s$ to calculate the difference of the point cloud generated by the network prediction and the truth label for classification of the point cloud, instead of using softmax normalization:

$$L_s\left(Q_p, F\right) = -\sum_{k=1}^{K} Q_P \delta\left(F\right)_k, \tag{9}$$

where $Q_p$ is the true value of point clouds of the label, and the $\delta$ represents the splicing of features of each category. The $F$ is the feature obtained by the feature extraction network in Eq. (4). The $k$ represents the categories of point cloud data set.

The feedback of loss function in feature vector can be defined as

$$L\left(\boldsymbol{F}_{p_i}\right) = \frac{1}{M}\sum_{m=1}^{M} n \left\| \boldsymbol{F}_{p_i} \cdot \boldsymbol{F}_{p_i}{}^T - \boldsymbol{E}_{\boldsymbol{F}_{p_i}} \right\|, \tag{10}$$

where $\boldsymbol{F}_{pi}$ is the fusion of features in $i$ scales. The $m$ is the times of the training, and the $M$ is the maximum times of

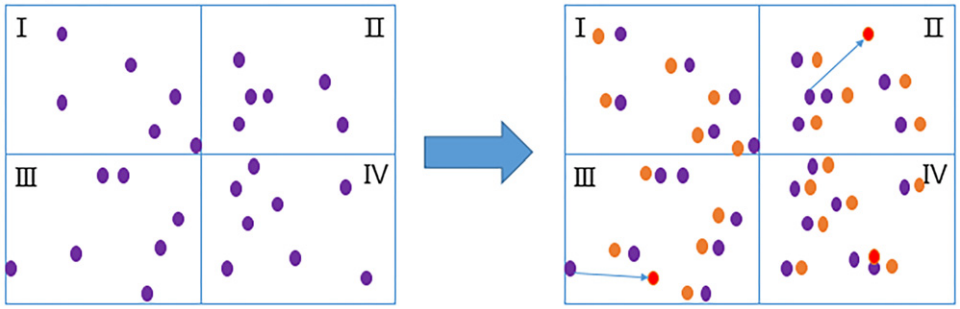



Figure 6. Schematic diagram of point cloud spatial shift upsampling.

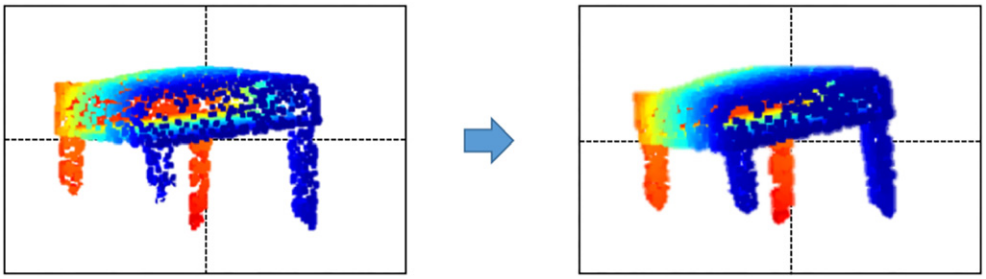


Figure 7. The difference of the table point cloud after spatial shift upsampling.

the training. The $\boldsymbol{F}_{pi}.\boldsymbol{F}_{pi}^{T}$ is the transposition multiplication of points, and the $\boldsymbol{E}_{\boldsymbol{F}_{pi}}$ is a diagonal matrix used to normalize the multiplied matrix.

The classification of the loss function in DiM-PCNet is defined as Eq. (11).

$$L_c\,(P, Q, F) = L_s\,(P, Q) + L\,(F) \times W, \qquad (11)$$

where $P$ is the label, $Q$ is the output, and $F$ is the features. $W$ changes with $i$ in Eq. (11), which is $W = \sum_{i=1}^{3} \frac{1}{i}$. The $W$ is the weight to control the influence of the feature on the loss function.

## 4. EXPERIMENT AND RESULT

The baseline in the experiment is PointNet, and the pyramid model is added. We test the DiM-PCNet on datasets of ShapeNet and ModelNet40. There are 16 different types of point cloud objects in ShapeNet. Its training set has 10240 point cloud files, and the test set has 4744 point cloud files. In ModelNet40, there are 40 different types of point cloud objects, in which there are 9843 point cloud files as training models and 2468 files as test models.

The hardware of the experiment is Intel E5-2683v3 (28 Cores 2.0 Ghz) +128 GB DDR4 ECC REG and NVIDIA TitanX GPU (GPU is provided by Yanshan University Supercomputing Center). The software environment is python3.5+pytorch.

### 4.1 *Parameter Setting and Performance Evaluation*

We use Adam as the experiment adaptive moment estimation optimizer for our model and preprocess the point cloud data before training. In order to select the appropriate training parameters, this experiment made corresponding parameter selection for epoch and batch-size.

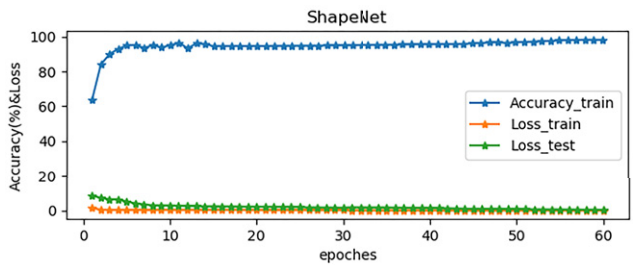



Figure 8. The accuracy and loss of ShapeNet with 60 epochs.

*(1) Epoch*

We adjust the parameters of network and sets different train epochs. The results of the train are shown in Figure 8, Figure 9. And the Figure 10, and Figure 11 are the accuracy of test on every class.

The training data set of ShapeNet is nearly 400 more than that of ModelNet40, and the test data set of ShapeNet is nearly half of that of ModelNet40. It can be seen from Figs. 8 and 9, the ShapeNet uses fewer epochs, and the accuracy is better than ModelNet40. Therefore, if the training sets have more quantity and each class is diverse, the model of the train will be more time-saving. Figs. 8 and 9 show that the loss of ShapeNet is stable between 50 and 60 epochs, so the learning has reached the best state. The loss of ModelNet40 also shows that there is a fluctuation among 120–140 epochs,

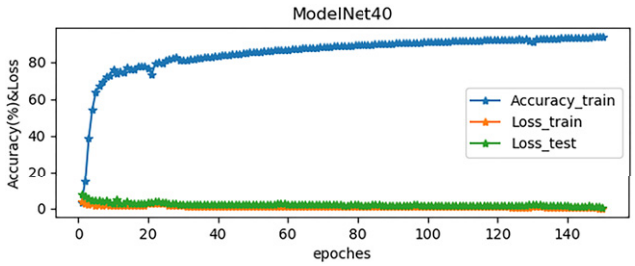



Figure 9. The accuracy and loss of ModelNet40 with 150 epochs.

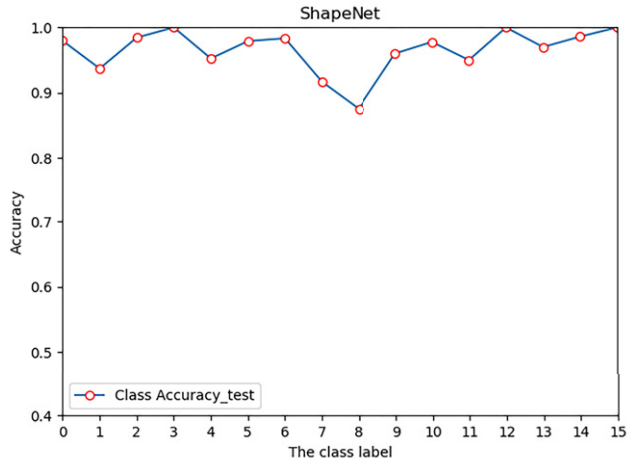



Figure 10. The test accuracy of ShapeNet class.

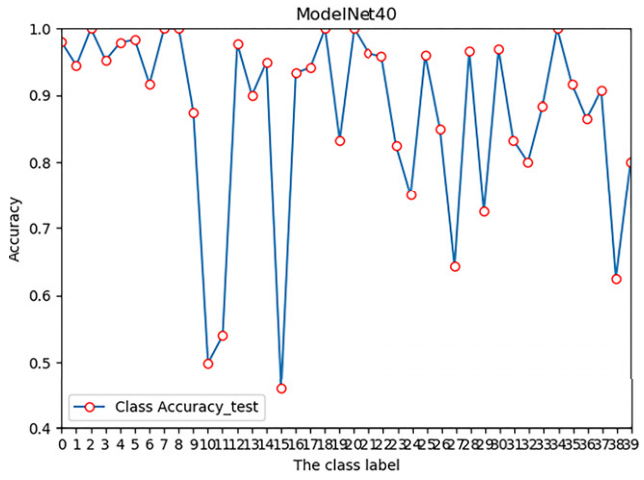



Figure 11. The test accuracy of ModelNet40 class.

and then it is stable among 140–150 epochs. Therefore, both ModelNet40 and ShapeNet are learning. It can be seen in Figs. 10 and 11, the accuracy of ShapeNet all over 80% and the overall trend fluctuates slightly, compared with ModelNet40.

*(2) Batch-size*

We use random shuffle on the datasets of ShapeNet and ModelNet40 to increase randomness and improve the network generalization ability. The position of shuffle dataset is not fixed, and the shuffle dataset is obtained by translation or rotation of the raw data.

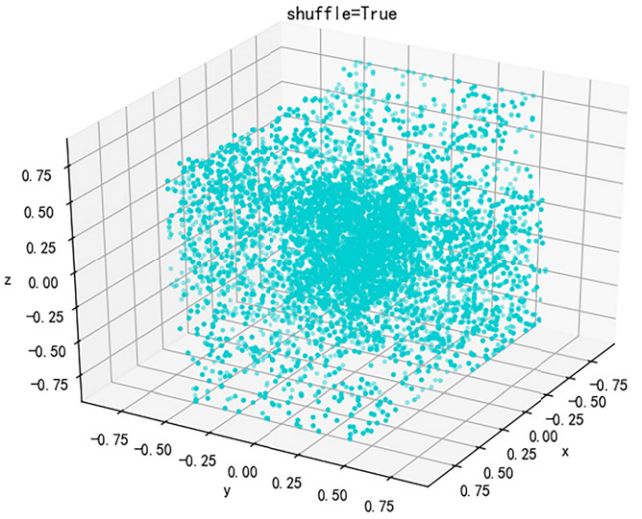



Figure 12. The spatial distribution of ModelNet4 with shuffle.

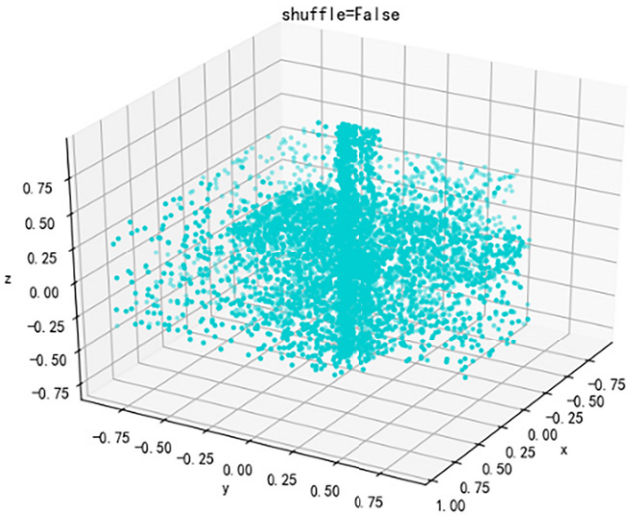



Figure 13. The spatial distribution of ModelNet4 without shuffle.

The Figures 12–15 are the visualization of an h5 training file randomly selected from ModelNet40 and ShapeNet, and we can see the results of shuffling in Table I.

As can be seen from Table I, the accuracy of ShapeNet with shuffling is 6.8% higher than that of the raw dataset. And the accuracy and the run time of ModelNet40 using shuffle are better than those of the raw dataset. So using the shuffle is effective in our model.

We set batch-size with 64 and the dropout with 0.4 to train our model. Eq. (13) is the evaluation of the accuracy in DiM-PCNet.

$$P_{OA} = \sum \frac{P_{\text{correct}}}{D_{\text{len}}}, \tag{12}$$

where $P_{\text{correct}}$ is the points of correct prediction, and the $D_{len}$ is the length of the datasets.

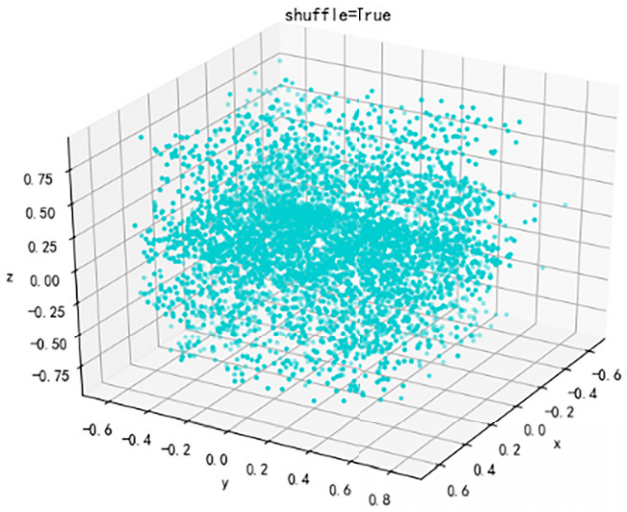



Figure 14. The spatial distribution of ShapeNet with shuffle.

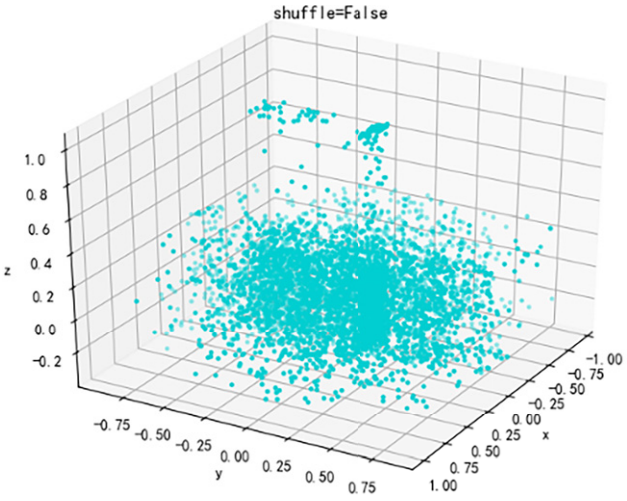



Figure 15. The spatial distribution of ShapeNet without shuffle.

**Table I.** Shuffle and raw datasets results (overall accuracy, %) on ModelNet40 and ShapeNet.

| Dataset | Input mode | Accuracy (%) | Runtime (min) |
|---|---|---|---|
| ModelNet40 | Shuffle | 86.9 | 286.8 |
| | raw | 85.7 | 289.5 |
| ShapeNet | Shuffle | 97.9 | 118.6 |
| | raw | 91.1 | 206.3 |

*(3) Downsampling*

We use the ModelNet40 to test the performance of the downsampling. The experiment shows that after the sampling operation, the accuracy of the DiM-PCNet is not significantly reduced, but the training is significantly accelerated. This is an ablation experiment to verify the effect of downsampling on the network.

**Table II.** Results of ablation experiment.

| | Accuracy (%) | Training time (min) | Average memory (MB) |
|---|---|---|---|
| Dim-PCNet with no downsample | 86.3 | 324.6 | 4853.48 |
| Dim-PCNet | 86.9 | 237.4 | 3287.20 |

It can be seen from the Table II that the accuracy is improved by 0.6% after downsampling, which is a very objective experimental result. And it can be explained that for multi-scale networks, the result of using downsampling is better than that of not using downsampling. Both training time and average memory usage are reduced. Because computer resources may be used by others, it is necessary to simplify the experimental data.

### 4.2 *Experimental Results*

#### 4.2.1 *Classification Experiment*

*(1) Accuracy Comparison for Data Sets*

The experimental results of ShapeNet and ModelNet40 are shown in Table III. The results of PointNet are carried out in the same experimental environment as the method in this paper.

We chose 11 categories of classical objects for comparison. For ShapeNet in Table III, the accuracy of our model is higher than PointNet in the earphone, chair, lamp, and guitar. It can be seen, the shape of the earphone, lamp, and guitar are all similar. When testing the accuracy of each category of ModelNet40 shown in Table III, we found that three categories' accuracy is reached to 100% by DiM-PCNet, while only one category achieved 100% with PointNet. Note that the accuracy difference of the car, cup, and table between our method and PointNet is 17.9%, 25%, and 39.5% respectively, which significantly improves the accuracy. The results show that our model can improve the accuracy of this kind of object with a regular shape. As can be seen in Table III, our method is 0.7% higher than PointNet for the mean accuracy of ShapeNet and is 1.4% higher than PointNet for ModelNet.

*(2) Accuracy Comparison of Different Methods*

The quantitative comparisons with the state-of-the art point-based methods are summarized in Table IV. Because the data set density of the three methods (DGCNN, DCG-Net, and SpiderCNN) is larger than ours, and machine learning is added to their methods, the accuracy of the three methods is about 0.8–1.5 percentage points higher than ours. Although the accuracy of our method does not outperform DGCNN, DCG-Net, and SpiderCNN on classification, combining the model complexity and robustness, our method shows good performance on point cloud recognition. On the whole, our method is the best among the methods of using MLP and is practical and feasible.

**Table III.** Classification results (accuracy, %) on ModelNet40 and ShapeNet dataset.

| Dataset | Category | PointNet | | DiM-PCNet | |
|---|---|---|---|---|---|
| | | Accuracy | Mean-accuracy | Accuracy | Mean-accuracy |
| ShapeNet | earphone | 72.2 | 97.2 | 85.7 | 97.9 |
| | airplane | 98.5 | | 93.0 | |
| | car | 96.6 | | 96.6 | |
| | chair | 98.2 | | 98.4 | |
| | lamp | 94.7 | | 94.8 | |
| | guitar | 99.0 | | 99.1 | |
| | table | 98.9 | | 98.9 | |
| ModelNet40 | bed | 100 | 85.5 | 95.0 | 86.9 |
| | bottle | 84.7 | | 84.7 | |
| | car | 82.1 | | 100 | |
| | cup | 75.0 | | 100 | |
| | dresser | 53.3 | | 90.0 | |
| | lamp | 75.0 | | 75.0 | |
| | table | 60.5 | | 100 | |

**Table IV.** Classification accuracy of DiM-PCNet and other models on ModelNet40.

| Method | Input | Accuracy (%) |
|---|---|---|
| PointNet | 2048 points | 85.5 |
| DiM-PCNet | 2048 points | 86.9 |
| ECC | 1024 points | 87.4 |
| PointNet | 1024 points | 89.2 |
| DiM-PCNet | 1024 points | 89.8 |
| PointNet++ | 1024 points | 90.7 |
| SPH3D-GCN | 1024 points | 90.8 |
| DiM-PCNet | 1024 points+normal | 91.4 |
| DGCNN | 1024 points | 92.2 |
| SpiderCNN | 1024 points+normal | 92.4 |
| DCG-Net | 1024 points | 93.1 |

*(3) Robustness of DiM-PCNet*

We set up 1, 10, 50, 100 Gaussian noise points in DiM-PCNet shown in Figure 16(a). As shown in Fig. 16(b), when the noise points are 10, the accuracy of PointNet can't reach a qualified classification standard, but ours still remains at 72%. When the noise points are 50, our method accuracy reaches 50%. Thought SpiderCNN is relatively stable and better than DiM-PCNet, when the noise points less than ten. As we can see, the accuracy of SpiderCNN with 50 noise points is less than 40%. With the increase of noise data, the classification effect will become worse. This shows that multi-scale and multi-layer are robust, and the robustness is relatively stable in our model. The accuracy trend chart in Fig. 16(b) shows the DiM-PCNet highlights great advantages in robustness. DiM-PCNet* in Fig. 16(b) represents the DiM-PCNet without direction consistency sampling. In addition, the adjustable parameters and the adaptable structure enhance the generalized capability of DiM-PCNet.

**Table V.** Complexity analysis of different method in classification.

| Method | Model size (MB) | Forward time (ms) | Accuracy (%) |
|---|---|---|---|
| PointNet | 13.3 | 35.2 | 89.2 |
| PointNet++ | 17.5 | 291 | 90.7 |
| DGCNN | 7.9 | 182.3 | 92.2 |
| SpiderCNN | 14 | 159.4 | 92.4 |
| DCG-NET | 23.8 | 203.2 | 93.1 |
| DiM-PCNet | 7.3 | 146.2 | 91.4 |

*(4) Model Complexity*

The results of Table V are the models on ModelNet40. We take the number $k$ of nearest neighbors as 20 in DGCNN, SpiderCNN, and SpiderCNN. The DiM-PCNet occupies the minimum memory, and forward time is also the second fastest. When the net uses KNN and graph convolution, the model will be more complex.

4.2.2 *Semantic Segmentation Experiment*

The experimental data set is S3DIS [35] that has 6 regions. We train the model with region 1-4. We evaluate the model used the region 5 of S3DIS. We used three metrics to evaluate the scene segmentation experiment, which is the average accuracy of each object (MAcc), the Mean Intersection over Union of all objects (Miou), and the overall accuracy (OA).

The accuracy for each class is shown in Eq. (13).

$$P_{acci} = \frac{TQ_i}{T_i}, \tag{13}$$

where $TQ_i$ is the correct semantic label, also the prediction label, and $T_i$ is the number of all labels.

$$mAcc = \frac{1}{N}\sum_{i=1}^{N} P_{acci} \tag{14}$$

mAcc is the average accuracy of each class, and $N$ is the number of classes of indoor scenes.

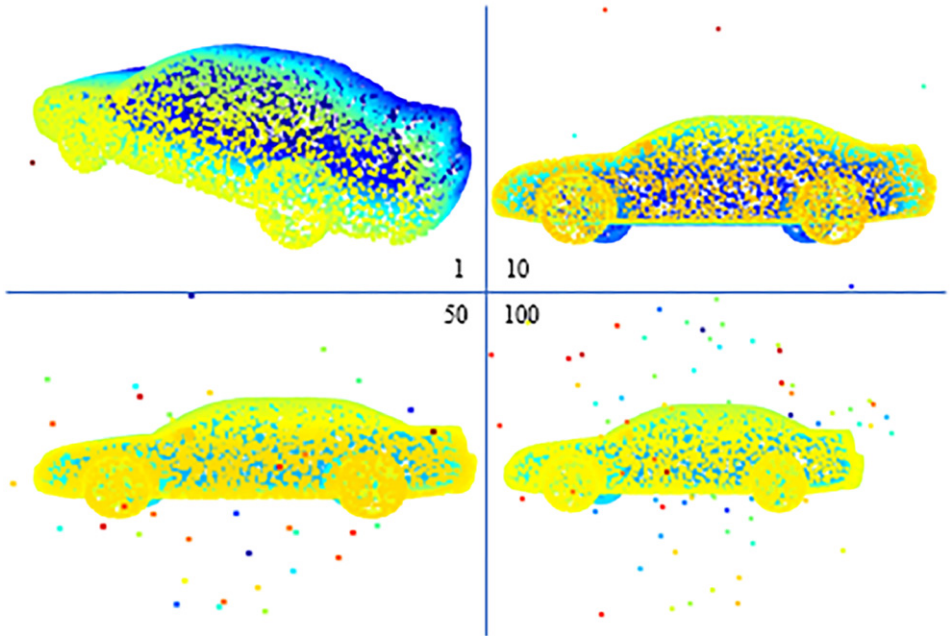
The Eq. (15) depicts the Iou of each class, and Eq. (16) depicts the mIou.

$$Iou_i = \frac{TQ_i}{T_i + Q_i - TQ_i} \tag{15}$$

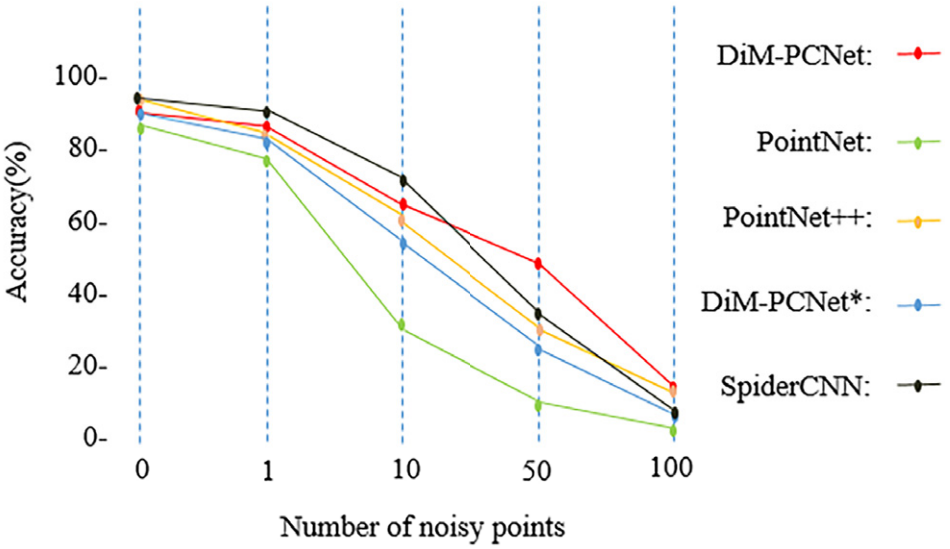
$$mIou_i = \frac{1}{N}\sum_{i=1}^{N} TQ_i/(T_i + Q_i - TQ_i). \tag{16}$$

The accuracy of the point cloud in the whole scene is shown in the Eq. (17).

$$OA = \frac{\sum_{b=1,n=1}^{b=B,n=N\_P} \mathrm{corr}(p_i)}{B\_S \times N\_P}. \tag{17}$$




**(a)The point cloud of car with different Gaussian noise.**




**(b) The robustness chart of our networkFIGURE**

Figure 16. The comparisons result with DiM-PCNet, PointNet, PointNet++, DiM-PCNet*, and SpiderCNN of the network robustness.

In Eq. (17), corr ($p_i$) represents the prediction accuracy of semantic tags of each point, $B\_S$ is the number of batches of the point cloud_ size, $N\_P$ is the points number of the point cloud for each batch_points.

In Table VII, our model has six categories that IOU accuracy reaches the best, but SPH3D-GCN has four categories. The six categories of our model contain the wall, column, table, window, beam, and board, which are the common and basic architectural elements in the indoor space. It proves the block of multi-scale and multi-level feature extraction improves the utilization of features.

Our model achieves a better average accuracy and overall accuracy of classes in Table VI. The multi-scale and multi-level strategy get the categories features, which applies the experiment of segmentation can also get better accuracy.

## 5. CONCLUSION

In this paper, we introduce the fusion of multi-scale and multi-level network (DiM-PCNet). The direction consistent sampling produces point clouds of different scales. In the DiM feature extraction block, features by the multi-level encoder generate high dimensional features that retain plenty

**Table VI.** The IOU of all categories on S3DIS.

| Method | PointNet | Segcloud [36] | SPG [37] | PointNet++ | SPH3D-GCN [38] | Our model |
|---|---|---|---|---|---|---|
| ceiling | 88.0 | 90.1 | 89.3 | 91.4 | 92.3 | 89.6 |
| floor | 95.7 | 96.1 | 96.8 | 95.2 | 98.2 | 96.1 |
| wall | 69.3 | 69.8 | 71.9 | 69.4 | 71.9 | 72.2 |
| beam | 0.05 | 0.0 | 0.0 | 00.0 | 0.03 | 0.1 |
| column | 23.1 | 18.4 | 15.0 | 16.2 | 17.6 | 24.3 |
| window | 46.2 | 38.4 | 46.5 | 66.1 | 46.8 | 47.5 |
| door | 51.6 | 23.1 | 61.5 | 14.8 | 43.8 | 58.8 |
| table | 52.6 | 70.5 | 69.4 | 70.3 | 71.0 | 71.2 |
| chair | 58.9 | 75.9 | 65.0 | 81.1 | 79.7 | 37.6 |
| sofa | 40.2 | 40.9 | 38.2 | 35.1 | 50.3 | 22.9 |
| bookcase | 5.9 | 58.4 | 52.6 | 57.7 | 32.0 | 39.4 |
| board | 26.4 | 12.9 | 2.1 | 50.4 | 25.8 | 26.7 |
| clutter | 33.2 | 41.6 | 51.3 | 51.4 | 52.7 | 36.6 |

**Table VII.** Comparison of segmentation results on S3DIS.

| Method | mAcc | mIou | OA |
|---|---|---|---|
| Segcloud | 57.35 | 48.92 | 80.80 |
| SPG | 64.40 | 54.10 | 82.90 |
| PointNet | 66.20 | 47.60 | 78.50 |
| PointNet++ | 66.85 | 54.90 | 86.43 |
| SPH3D-GCN | 67.06 | 57.20 | 86.94 |
| Our model | 67.20 | 53.70 | 87.10 |

of information provided by raw datasets, the representative, and detailed features of the point cloud. The best accuracy of classification is 91.4%, and the best accuracy of segmentation is 87.1% by the debugging in experimental parameters. It is worth noting that the robustness of DiM-PCNet reaches a stable state.

The multi-scale and multi-level feature extraction part can be used as a feature extraction tool independently, which can be embedded into any other point cloud classification networks conveniently. In the future, a larger amount of point clouds is likely to be produced and the point cloud annotations for training will become increasingly complex and time-consuming. In future work, the unsupervised network will be applied in DiM-PCNet to improve its applicability.

## ACKNOWLEDGMENT

This work was supported in part by Department of Education of Hebei Province under Grant ZD2020176, Hebei Science and Technology Department project 21470302D.

## REFERENCES

[1] W. Peng, H. Pan, H. Liu, and Y. Sun, "IDA-3D: Instance-depth-aware 3D Object detection from stereo vision for autonomous driving," *Proc. CVPR, Seattle, WA, USA* (IEEE, Piscataway, NJ, 2020), pp. 13012–13021.

[2] S. Han and Z. Xi, "Dynamic scene semantics SLAM based on semantic segmentation," IEEE Access **8**, 43563–43570 (2020).

[3] J. Ping, Y. Liu, and D. Weng, "Comparison in depth perception between Virtual Reality and Augmented Reality Systems," *Proc. VR* (IEEE, Piscataway, NJ, 2019), pp. 1124–1125.

[4] Y. Li, G. Tong, X. Li, L. Zhang, and H. Peng, "MVF-CNN: Fusion of multilevel features for large-scale point cloud classification," IEEE Access **7**, 46522–46537 (2019).

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," IEEE Trans. Pattern Anal. Mach. Intell. **37**, 1904–1916 (2014).

[6] A. Bucksch and K. Khoshelham, "Localized registration of point clouds of botanic trees," IEEE Geosci. Remote Sens. Lett. **10**, 631–635 (2013).

[7] C. Wang, M. Wu, Z. Wang, L. Wang, H. Sheng, and J. Yu, "Neural opacity point cloud," IEEE Trans. Pattern Anal. Mach. Intell. **42**, 1570–1581 (2020).

[8] C. Xiang, C. R. Qi, and B. Li, "Generating 3D adversarial point clouds," *Proc. CVPR* (IEEE, Piscataway, NJ, 2019), pp. 9128–9136.

[9] J. Hou, "Permuted sparse representation for 3D point clouds," IEEE Signal Process. Lett. **26**, 1847–1851 (2019).

[10] J. Li, Y. Bi, and G. H. Lee, "Discrete rotation equivariance for point cloud recognition," *Proc. ICRA* (IEEE, Piscataway, NJ, 2019), pp. 7269–7275.

[11] L. Zhang, J. Sun, and Q. Zheng, "3D point cloud recognition based on a multi-view convolutional neural network," Sensors **18**, 3681 (2018).

[12] E. Kalogerakis, M. Averkiou, S. Maji, and S. Chaudhuri, "3D shape segmentation with projective convolutional networks," *Proc. CVPR* (IEEE, Piscataway, NJ, 2017), pp. 6630–6639.

[13] C. R. Qi, H. Su, M. Nießzner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," *Proc. CVPR* (IEEE, Piscataway, NJ, 2016), pp. 5648–5656.

[14] M. Liang, B. Yan, S. Wang, and R. Urtasun, "Deep continuous fusion for multi-sensor 3D object detection," *Proc. European Conf. on Computer Vision (ECCV)* (Springer, Cham, 2018), Vol. 11220, pp. 663–678.

[15] F. Poux and R. Billen, "Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring versus deep learning methods," ISPRS Int. J. Geo-Inf. **8**, 213 (2019).

[16] C. Wang, M. Cheng, F. Sohel, M. Bennamoun, and J. Li, "NormalNet: A voxel-based CNN for 3D object classification and retrieval," Neurocomputing **323**, 139–147 (2019).

[17] J. Huang and S. You, "Point cloud labeling using 3D convolutional neural network," *Proc. Int'l. Conf. Pattern Recognit. (ICPR)* (IEEE, Piscataway, NJ, 2016), pp. 2670–2675.

[18] M. Daniel and S. Sebastian, "VoxNet: A 3D convolutional neural network for real-time object recognition," *Proc. IEEE/RSJ* (IEEE, Piscataway, NJ, 2015), pp. 922–928.

[19] C. R. Qi, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," *Proc. CVPR* (IEEE, Piscataway, NJ, 2017), pp. 77–85.

[20] R. Huang, D. Hong, Y. Xu, W. Yao, and U. Stilla, "Multi-scale local context embedding for lidar point cloud classification," IEEE Geosci. Remote Sens. Lett. **17**, 721–725 (2020).

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," *Proc. NIPS* (2018), pp. 5105–5114.

[22] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," ACM Trans. Graph. **38**, 1–12 (2019).

[23] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," *Computer Vision–ECCV 2018. ECCV 2018*, Lecture Notes in Computer Science (Springer, Cham, 2018), Vol. 11212, pp. 87–102.

[24] A. T. Melekhov, T. Sattler, M. Pollefeys, E. Rahtu, and J. Kannala, "DGC-Net: Dense geometric correspondence network," *2019 IEEE Winter Conf. on Applications of Computer Vision (WACV)* (IEEE, Piscataway, NJ, 2019), pp. 1034–1042.

[25] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2017).

[26] R. Chen, S. Han, J. Xu, and H. Su, "Point-based multi-view stereo network," *Proc. ICCV* (IEEE, Piscataway, NJ, 2019), pp. 1538–1547.

[27] F. Engelmann, T. Kontogianni, A. Hermans, and B. Leibe, "Exploring spatial context for 3D semantic segmentation of point clouds," *Proc. ICCVW* (IEEE, Piscataway, NJ, 2017), pp. 716–724.

[28] X. Zhang, K. Zhu, G. Chen, X. Tan, L. Zhang, F. Dai, P. Liao, and Y. Gong, "Geospatial object detection on high resolution remote sensing imagery based on double multi-scale feature pyramid network," Remote Sens. **11**, 755 (2019).

[29] Z. Zhao, Y. Cheng, X. Shi, and X. Qin, "Terrain classification of liDAR point cloud based on multi-scale features and PointNet," Laser & Optelectronics Progress **56**, 251–258 (2019).

[30] J. Bai and H. Xu, "MSP-Net: Multi-scale point cloud classification network," J. Comput.-Aided Design & Computer Graphics **2019**, 1917–1924.

[31] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan, "DensePoint: Learning densely contextual representation for efficient point cloud processing," *Proc. ICCV* (IEEE, Piscataway, NJ, 2019), pp. 5238–5247.

[32] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "PointASNL: Robust point clouds processing using nonlocal neural networks with adaptive sampling," *Proc. CVPR* (IEEE, Piscataway, NJ, 2020), pp. 5588–5597.

[33] Y. Eldar, M. Lindenbaum, M. Porat, and Y. Y. Zeevi, "The farthest point strategy for progressive image sampling," IEEE Trans. Image Process. **6**, 1305–1315 (1997).

[34] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," *in Proc. CVPR* (IEEE, Piscataway, NJ, 2020), pp. 11105–11114.

[35] O. S. Armeni, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese, "3D semantic parsing of large-scale indoor spaces," *Proc. IEEE Int'l. Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2016), pp. 6–7.

[36] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese, "SEGCloud: Semantic segmentation of 3D point clouds," *2017 Int'l. Conf. on 3D Vision (3DV)* (IEEE, Piscataway, NJ, 2017), pp. 537–547.

[37] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," *2018 IEEE/CVF Conf. on Computer Vision and Pattern Recognition* (IEEE, Piscataway, NJ, 2018), pp. 4558–4567.

[38] H. Lei, N. Akhtar, and A. Mian, "Spherical kernel for efficient graph convolution on 3D point clouds," IEEE Trans. Pattern Anal. Mach. Intell. **43**, 3664–3680 (2021).