Point Cloud Classification Network Based on Dynamic Graph Convolution

Ke Wu, Hong Dai*, Shuang Wang, and Chengrui Liu

Abstract—**With the continuous development of 3D data acquisition technology, point cloud data, an essential form of 3D data representation is widely used in fields such as autonomous driving, indoor navigation, virtual reality, and augmented reality. As an essential branch of point cloud data processing, cloud classification has important research significance and value. However, this task poses challenges due to the sparsity, irregularity, and unordered nature of point cloud data. Among most of the methods dealing with this problem, there are problems of inadequate extraction of local features, low accuracy of classification, and poor generalization of point cloud data. Therefore, this paper addresses the above issue. This paper presents an improved 3D point cloud classification network by enhancing the Dynamic Graph Convolutional Neural Network (DGCNN). Firstly, this paper combines K-Nearest Neighbors (KNN) and ball radius query for feature extraction to better capture the local structural information in cloud classification data. Secondly, a conventional convolution layer is incorporated between the second and third graph convolution layers to enhance the feature representation in the proposed approach. Finally, this paper uses a global pooling method that combines maximum pooling and average pooling to construct the global structure of the point cloud while preserving the most critical information. Realize 3D point cloud classification. The experimental results demonstrate a 0.9% improvement in accuracy on the ModelNet40 dataset using the proposed improved method. This validates the effectiveness of the enhancement process described in this paper and offers valuable insights for enhancing the performance and application of point cloud classification.**

Index Terms—**Point cloud classification, Dynamic graph convolution, Feature extraction, Local structure information**

I. Introduction

In recent years, with the improvement of 3D scanning technology and point cloud data acquisition, point cloud data has been widely used in 3D scene recognition, robot navigation, and autonomous driving [1]. As the most direct expression of 3D information, point cloud data processing has always been concerned. Therefore, as a basic problem of point cloud data processing, point cloud classification has been a hot research topic. It refers to assigning a fixed point cloud data set and a set of labels to classify the point cloud data into different categories. Traditional point cloud classification methods usually use hand-designed feature extractors and classifiers. For example, HKS [2], Spin Images(SI) [3], 3D Shape Descriptor(3DSD) [4] and other methods. However, these methods have limitations that result in not fully exploiting the features of the point cloud data. Point cloud classification methods based on neural networks have significantly progressed in recent years. For example, PointNet [5] is the first completely neural network-based point cloud classification method. It can receive the point cloud directly as input without preprocessing. PointNet++ [6], an extended version of PointNet, proposes a multi-scale architecture better to capture local and global features in point clouds. DGCNN [7] is a point cloud classification method based on Graph Convolutional Neural Network (GCNN) [8], which can better capture the features of point cloud data. The development of these neural network-based point cloud classification methods has significantly improved the accuracy of point cloud classification and has played an important role in practical applications.

DGCNN is a classical point cloud classification method based on a neural network. It extracts the features of point cloud data and classifies them through local feature aggregation and global feature coding. In local feature aggregation, DGCNN uses KNN query [9] to obtain local information of neighboring points and then aggregates these local features through a fully connected layer. DGCNN uses pooling operations in global feature coding to extract global features from point cloud data. Although DGCNN has achieved good performance in point cloud classification tasks, it also has some problems. First, the number of neighbor points of the KNN query is not enough to capture the global information of the point cloud data, resulting in low classification accuracy. Secondly, DGCNN uses maximum pooling operation to extract global features of point cloud data, which may ignore some vital feature information, resulting in inaccurate feature extraction. Therefore, this paper's proposed method combines the spherical radius query [10] and the KNN query. It incorporates a traditional convolution layer for multi-scale fusion and employs a global pooling method to extract features. The improved model

Manuscript received May 13, 2023; revised September 5, 2023. This work is supported by Graduate Education Reform and Innovation Project, University of Science and Technology Liaoning (LKDYC202221).

Ke Wu is a Postgraduate of the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (e-mail: 786854814@qq.com).

Hong Dai* is a Professor of the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, China. (corresponding author to provide phone: +086-186-4226-8599; fax: 0412-5929818; e-mail: dear_red9@163.com).

Shuang Wang is a Postgraduate of the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, CO 114051, China. (e-mail: 1657669526@qq.com).

Chengrui Liu is a Postgraduate of the School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, Liaoning, CO, 114051, China. (e-mail: cherryliumark@163.com).

exhibits superior performance in 3D point cloud classification tasks.

II. Related Work

Point cloud classification is an essential problem in computer vision, which aims to classify point cloud data into different categories. It has become a research hotspot [11], and many neural network-based methods have emerged.

Traditional point cloud classification is performed by manually designing a series of features and then directly classifying the 3D point cloud using a suitable classifier [12]. However, although manual feature extraction has achieved some results, the disadvantages are also prominent. The feature extraction process overlooks the inter-correlation between features [13], thereby requiring extensive experimentation to identify the optimal combination of features. This requires domain experts with relevant domain knowledge, intense subjectivity, and weak adaptability to complex point cloud scenarios. Deep learning algorithms have powerful feature extraction and representation capabilities [14]. Features can be automatically learned from the original point cloud data, avoiding manual feature design's tedious and subjective nature. Due to the unstructured nature of point cloud data [15], the advancement of hardware facilities has facilitated the use of deep learning techniques. These techniques enable the extraction of latent information from data and effectively tackle complex classification tasks, thereby enhancing the accuracy of cloud classification [16-17]. In recent years, the development of deep learning in the image field has led researchers to apply deep learning to point cloud classification.

In order to better apply deep learning to point cloud classification, early deep learning technologies projected 3D point clouds into multiple views or voxel grids. Su et al. [18] proposed a Multi-View Convolutional Neural Network (MVCNN) to develop a view-based 3D shape feature extraction method. This article projects the mesh surface onto multi-view images and extracts features separately. Then, fuse and classify the learned information in the pooling layer. VoxNet proposed by Maturana et al. [19], provides a compelling point cloud classification technique by converting point cloud data into 3D normalized voxel data and using convolutional neural networks to learn features and perform classification. However, although view-based or voxel-based methods have made some progress in processing point cloud data, they are more computationally intensive and, at the same time, insufficient for data feature extraction. Therefore, using deep learning to process point cloud data directly has become a current research hotspot.

In 2017, Charles et al. [5] of Stanford University proposed PointNet, the first deep-learning network that can directly handle the classification of disordered point clouds. This method independently processes the data of each point cloud through Multi-Layer Perceptron (MLP) [20]. Meanwhile, the Spatial Transformation Network (STN) [21] is also used to solve the point cloud data rotation-invariance problem and the symmetric function of maximum pooling (Maxpooling) to solve the point cloud data disorder [22]. However, the disadvantages are apparent enough. This method only focuses on the global information of the point cloud data. It fails to utilize the local information fully, thus limiting the ability to extract deeper features and losing the universality of complex scenes. Therefore, to solve the above problems, the team proposed an improved version of PointNet, PointNet++ [6]. To introduce local information based on PointNet, one can use the Farthest Point Sampling (FPS) algorithm to construct local neighborhoods and group the sampled points using the ball radius query method. However, feature extraction still employs independent point convolution and does not establish the relationship among points. Therefore, Wang et al. [7] proposed a Dynamic Graph Convolutional Neural Network (DGCNN). The core structure of the network is the EdgeConv layer. Edge convolution captures the relationship between centroids and their edge points. It can create local neighborhood maps to extract local features better. The network establishes the connection between points and local neighborhoods, effectively improving the accuracy of point cloud classification. However, DGCNN still has some drawbacks. The method's focus on the distance relationship among points during the construction of the local neighborhood is limited in capturing the local features of the point cloud. As a result, the method exhibits poor feature expression ability and lacks robustness.

Based on the above analysis, this paper improves on the problems of DGCNN and proposes a new 3D point cloud classification network. First, combining KNN and ball radius query in feature extraction makes extracting local features more adequate based on DGCNN edge convolution. Then, the traditional convolution layer is added to improve the depth of the model, which can help the model better adapt to the data and enhance the ability of model feature characterization. Finally, this paper constructs a global pooling method by combining maximum pooling and average pooling, which serves as the primary approach for extracting global features. This improves the robustness of the model and reduces the risk of overfitting compared to the previous approach. Due to fully considering both the local and global characteristics of point cloud data, the proposed model achieves superior classification performance compared to the baseline model. On the ModelNet40 dataset, the proposed model achieves an accuracy of 93.8%. Compared with other point cloud classification methods, the method in this paper performs well in terms of accuracy and stability. The improved method in this paper has important implications for point cloud classification and related fields. It can offer more accurate and efficient solutions for practical applications.

III. Build Model

A. *Network Structure Design*

This paper designs a point cloud classification network with an unordered point set as the input. The network's input is the 3D point set coordinates (x, y, z), where only the coordinate positions are considered, not the normal vectors, colors, and other factors. The original point cloud is input, transformed, and aligned by the Spatial Transform Network (STN), and then, the Dynamic Graph Convolutional (DGC) operation is used to obtain the local features. As shown in Fig. 1, the classification network contains four layers of convolutional blocks. From left to right, it first passes through the graph convolution with 64 and 128 convolution

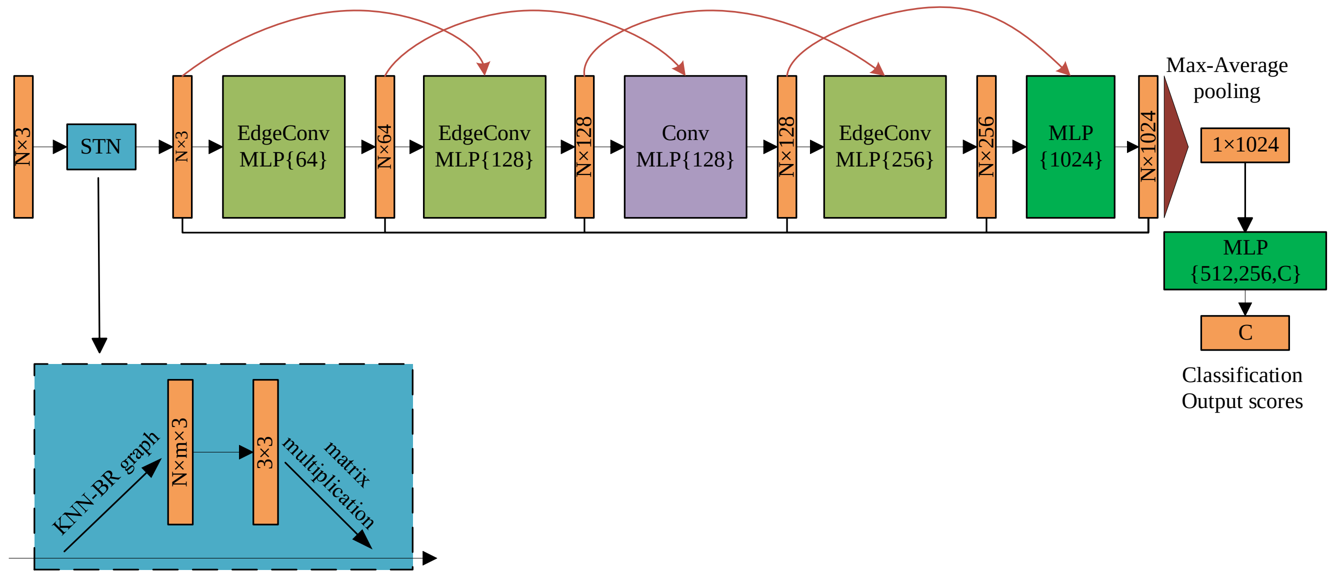

Fig. 1 Model architectures

kernels, then the traditional convolution with 128 convolution kernels, and finally, the graph convolution with 256 convolution kernels. In the feature extraction process, one can fuse the previous layer's local features again while extracting local features at a deeper level. Finally, the feature output of each layer is united, and then the global feature is extracted through MLP.

B. Dynamic Graph Convolution

Graph Convolutional Neural Network (GCNN) [8] is a neural network model based on graph structure. It realizes the representation learning of graph data by aggregating and summarizing the nodes' and neighbors' information. Traditional graph convolution applies a fixed graph to each layer in the network, and convolution operations are applied to the graph structure. Therefore, convolution calculation is based on a fixed adjacency matrix. Dynamic graph convolution is a convolution calculation method based on a dynamic adjacency matrix [24]. The dynamic graph convolution used in this paper improves EdgeConv in DGCNN, in which the KNN query is improved to a combination of KNN query and ball radius query. After querying and calculating distances, one can construct a dynamic adjacency matrix and divide feature extraction using dynamic graph convolution into two steps. (1) Construct the graph structure using KNN and ball radius query algorithm. (2) Extract the features of edges in the graph structure using MLP.

i. Building Feature Structure Extraction Module of Structure-Graph

In point cloud data, the density of each point is not necessarily the same, so the number of neighbors of each point may vary significantly in a traditional KNN query. Some points may have more neighboring points, while others may have only a small number of neighboring points. In this case, using the same *K* value may not fully exploit the topology of the point cloud data and thus affect the model's performance. Therefore, combining KNN query and ball radius query can better capture the local structure of point cloud data, leading to improved model robustness and generalization ability.

For a given center point p_i, the K-nearest neighbors algorithm is applied to select k neighboring nodes $N_k(p_{ij})$, where $j = 1, 2, ..., k$. Equation (1) defines the graph structure $N_k(p_{ij})$ constructed by k.

$$N_k(p_{ij}) = \{p_{ij} \in p \mid j \in \arg\min_{k=1}^{K} |p_{ik} - p_i|^2\} \tag{1}$$

Subsequently, we employ the ball radius query algorithm to obtain all points within a specified radius r, denoted as $N_r(p_{ij})$, where $j = 1, 2, ..., r$. Equation (2) defines the graph structure $N_k(p_{ij})$ constructed by r.

$$N_r(p_{ij}) = \{p_{ij} \in p \mid p_{ij} - p_i| \le r\} \tag{2}$$

To explain the generation process of $N(p_{ij})$, the fusion of the neighborhoods $N_k(p_{ij})$ and $N_r(p_{ij})$ of p_i is described in Equation (3).

$$N(p_{ij}) = N_k(p_{ij}) \bigcup N_r(p_{ij}) = \begin{cases} N_k(p_{ij}), (k \ge r) \\ N_r(p_{ij}), (k < r) \end{cases} \tag{3}$$

After the above operation, all the neighboring points of the center point that meet the requirements are found by the combination of KNN and ball radius query. Assuming that there are m points and then connecting the central point p_i with all its neighbors p_{ij} through directed edges, Equation (4) to Equation (6) constructs the graph structure.

$$G = (V, E) \tag{4}$$

$$V = \{p_i \mid i = 1, 2, ..., N\} \tag{5}$$

$$E = \{e_i = (e_{i1}, e_{i2}, ..., e_{im} \mid i = 1, 2, ..., N)\} \tag{6}$$

In the graph structure G, V represents the set of N central nodes. E represents the directed edges formed by the central point and its neighbors. The edge feature between the central point p_i and the *m-th* neighbor point p_{im} is denoted as e_{im}. The features of each point change across different feature extraction layers, resulting in a dynamic neighborhood map in the feature space. This enables for continuous updates to the local structure of the point cloud as the network layers deepen, enhancing the network's ability to capture local information. After the above operation, the construction of the graph structure by KNN and ball radius query algorithm is completed, as Fig. 2 shows the graph structure constructed in this paper.

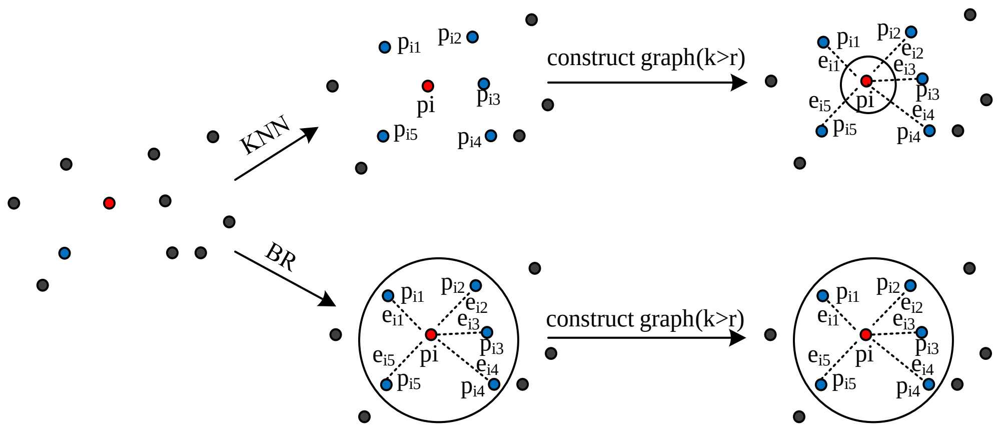

Fig. 2 Using KNN-BR to construct the graph structure

ii. MLP Feature Extraction

In this paper, the feature extraction of graph structure is realized using a shared multi-layer perceptron. In constructing the graph structure, all neighboring points p_{ij} of the center point p_i are queried for the MLP operation. For each point p_i, build the adjacency matrix A, where $A_{ij}=1$ means that point p_{ij} is a neighbor point of point p_i. Otherwise, it is 0. Equation (7) shows the constructed adjacency matrix A_{ij}.

$$A_{ij}=\begin{cases}1, & p_{ij}\in N(p_{ij})\\ 0, & otherwise\end{cases} \tag{7}$$

For each point p_i and its neighbor point p_{ij}, calculate the weight w_{ij}. (a) The weights w_{ij} can be calculated directly using the Euclidean distance for KNN neighbor points whose p_{ij} is p_i. (b) For p_{ij} is a spherical radius neighbor point of p_i, it is necessary to now calculate the relative distance ρ_{ij} from the distance between p_i and p_{ij} as shown in Equation (8). The weights w_{ij} are then computed using the Gaussian function, where σ is the learnable parameter, so Equation (9) shows the weights w_{ij}.

$$\rho_{ij}=\frac{|p_{ij}-p_i|}{r} \tag{8}$$

$$w_{ij}=\begin{cases}\exp(-|p_{ij}-p_i|^2), & p_{ij}\in N_k(p_{ij})\\ \exp(-\frac{\rho_{ij}^2}{2\sigma^2}), & p_{ij}\in N_r(p_{ij})\end{cases} \tag{9}$$

After calculating the weights, one needs to nonlinearly transform the feature vector of each neighbor point and then obtain the updated feature vector after weighted aggregation. Specifically, define the feature vector f_i of each point p_i as its three-dimensional coordinates (x_i, y_i, z_i), and thus the feature vector of the neighbor node p_{ij} is f_{ij}. Then, use a fully connected layer as MLP to nonlinearly transform the feature vector of the neighbor points, obtaining the hidden layer feature vector h_{ij} as shown in Equation (10).

$$h_{ij}=RELU(W_1 f_{ij}+b_1) \tag{10}$$

Where W_1 is the weight matrix of the fully connected layer, b_1 is the bias vector. Utilizing the *RELU* activation function [25] in neural networks enables better adaptation to nonlinear modeling. It effectively captures nonlinear relationships, particularly in addressing complex point cloud problems, while enhancing the model's generalization ability. Then, we weigh and aggregate the feature vector h_{ij} and weight w_{ij} of the hidden layer to obtain the new feature vector f'_{ij} of point p_{ij} as shown in Equation (11).

$$f'_{ij}=RELU(W_2\textstyle\sum_{j\in m} w_{ij}h_{ij}+b_2) \tag{11}$$

W_2 is the weight matrix of the second layer of MLP, and b_2 is the bias vector. After obtaining the new feature vector f'_{ij}, the feature vectors of all points are divided into t groups, each containing n feature vectors. We fuse the feature vectors for each group using the maximum pooling and average pooling operations to receive a new set of feature vectors g_{ij}. Finally, all groups of feature vectors are connected to obtain the global feature vector G of the whole point cloud. Equation (12) shows the specific formula.

$$g_{ij}=\frac{1}{n}\textstyle\sum_{j\in m} f'_{ij}+\max\sum_{j\in m} f'_{ij},\quad G=[g_1,g_2,...,g_t] \tag{12}$$

This leads to the core modules in the network structure of this paper, as shown in Fig. 3. Learning features in local graph structures using graph convolution. N represents the number of input points. e represents the feature dimension of each input point. Here e=3, m represents the number of neighbor points of the point cloud centroid, {*32, 64, …, D*} represents the number of neurons in each perceptual layer of the MLP, and the neuron in the last layer is D. Therefore, the feature dimension obtained after the pooling operation for feature aggregation is $N\times D$.

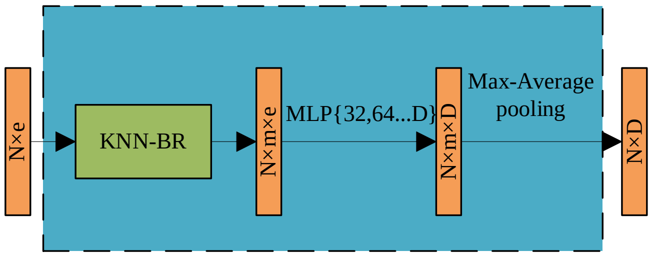

Fig. 3 Feature extraction module of graph structure

C. Spatial Transformation Network

PointNet proposes that the function of the Spatial Transformation Network (STN) is to train a set of spatial rotation matrices. This rotation matrix can coordinate the alignment of the input point cloud data. The spatial transformation network can directly rotate the point cloud

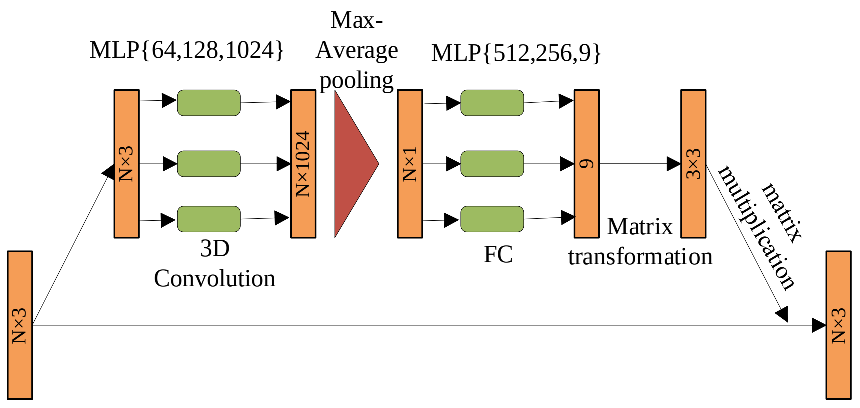

Fig. 4 Spatial transformation network

data to a better angle, which is more beneficial for the network to classify the data. The specific implementation is shown in Fig. 4. Multiple MLPs align the input cloud data with a maximum pooling prediction 3*3 rotation matrix, multiplied directly with the input point cloud to achieve coordinate alignment.

IV. EXPERIMENTS

A. Experimental Dataset and Evaluation Index

i. Experimental Dataset

The dataset used for the 3D point cloud classification experiments in this paper is the ModelNet40 dataset [26]. The Princeton University 3D Vision Organization provides the dataset and has 12, 311 meshed CAD models of manufactured objects. A distinctive feature of this dataset is its diversity, containing 40 categories that can cover a large number of real-world 3D objects. There are 9843 models in the dataset for training and 2468 models for testing. As shown in Table I, the number of samples used for training and testing is given for the 40 categories. Another essential feature of this dataset is its challenging nature. Since the models in the dataset come from different categories, with further deformation and scaling, the algorithm must be able to distinguish these objects accurately.

ii. Evaluation Index

The evaluation metrics provide an adequate and intuitive assessment of the effectiveness of the point cloud classification model. Among several metrics, such as accuracy, spatial complexity, and execution time, the the accuracy metric is the most critical. This paper selects the most commonly used accuracy metrics for evaluating point cloud classification tasks. The overall accuracy (*OA*), precision, and mean accuracy (*mA*), as expressed in Equations (13) to (15), are used to evaluate the performance in our study.

TABLE I
EXPERIMENTAL DATA

Classification	Number of training samples	Number of testing samples	Classification	Number of training samples	Number of testing samples
Airplane	626	100	Bathtub	106	50
Bed	515	100	Bench	173	20
Book shell	572	100	Bottle	335	100
Bowl	64	20	Car	197	100
Chair	889	100	Cone	167	20
Cup	79	20	Curtain	138	20
Desk	200	86	Door	109	20
Dresser	200	86	Flower pot	149	20
Glass box	171	100	Guitar	155	100
Keyboard	145	20	Lamp	124	20
Laptop	149	20	Mantel	284	100
Monitor	465	100	Night stand	200	86
Person	88	20	Piano	231	100
Plant	240	100	Radio	104	20
Range hood	115	100	Sink	128	20
Sofa	680	100	Stairs	124	20
Stool	90	20	Table	392	100
Tent	163	20	Toilet	344	100
Television stand	267	100	Vase	475	100
Wardrobe	87	20	Xbox	103	20

$$OverallAccuracy = \frac{TP+FN}{TP+FP+TN+FN} \quad (13)$$

$$Precision = \frac{TP}{TP+FP} \quad (14)$$

$$MeanAccuracy = \sum_{m=1}^{M} Precision_m / M \quad (15)$$

True positive (*TP*) indicates the model correctly predicts positive cases. False positive (*FP*) shows the model incorrectly predicts negative cases to be positive cases. False negative (*FN*) suggests the model incorrectly predicts a positive case to be a negative case. True negative (*TN*) indicates the model has correctly predicted the negative instances. *M* represents the number of categories of the classification.

B. Experimental Results

The test results of different network models for the ModelNet40 dataset are shown in Table II. Overall Accuracy （*OA*） in the table indicates the overall classification accuracy, the result of correct classification of all tested several models. Mean Accuracy (*mA*) represents the average classification accuracy and the mean of the accurate results for each category in the test set. The comparison shows that the model proposed in this article achieves significant results in *OA* and *mA* metrics on the ModelMet40 dataset compared to DGCNN and other classic models, proving that the model performs better.

TABLE II
POINT CLOUD CLASSIFICATION RESULTS

Model	Input	Points	OA	mA
VoxNet	voxel	-	85.9	83.0
PointNet	xyz	1024	89.2	86.0
PointNet++	xyz	1024	90.7	-
PointNet++(msg)	xyz, normal	1024	92.9	90.4
PointCNN	xyz	1024	92.2	88.1
PointConv	xyz, normal	1024	92.5	-
MVCNN	image	-	90.1	-
DGCNN	xyz	1024	92.9	90.2
Ours	**xyz**	**1024**	**93.8**	**90.6**

Compared with the VoxNet and MVCNN algorithms, the algorithm in this paper deals with each point in the point cloud directly. It can perceive local fine-grained structural information and reduce the information loss caused by voxelization and multi-view conversion process, so the accuracy rate increases by 4.3%~8.5%. Compared with PointNet and PointNet++, the algorithm in this paper considers the set association between points when obtaining single-point features. Local information descriptions are enhanced to facilitate detailed feature mining, improving accuracy by 2.5%~5.2%. Compared with DGCNN and other algorithms, the improved algorithm in this paper enhances feature linking between dynamic graphs. It has gained a deeper understanding of the intrinsic correlations in feature maps, resulting in an accuracy improvement of 0.9%.

Fig. 5 plots the changes in *OA* during the training of the proposed model in this paper and the baseline model. As can be seen from the figure, compared with the DGCNN network model under the same conditions, the *OA* of the whole process of the proposed model in this paper is significantly higher than that of DGCNN. The model's accuracy stabilizes after 150 iterations, while the DGCNN model needs the required stability after 200 rounds. Figure 6 depicts the variation in mA throughout the training process, while both demonstrate improved experimental outcomes. They illustrate the proposed model's accuracy, displaying slight fluctuations and consistent enhancements over time. It shows that the proposed model in this paper has a more vital local feature extraction ability than DGCNN.

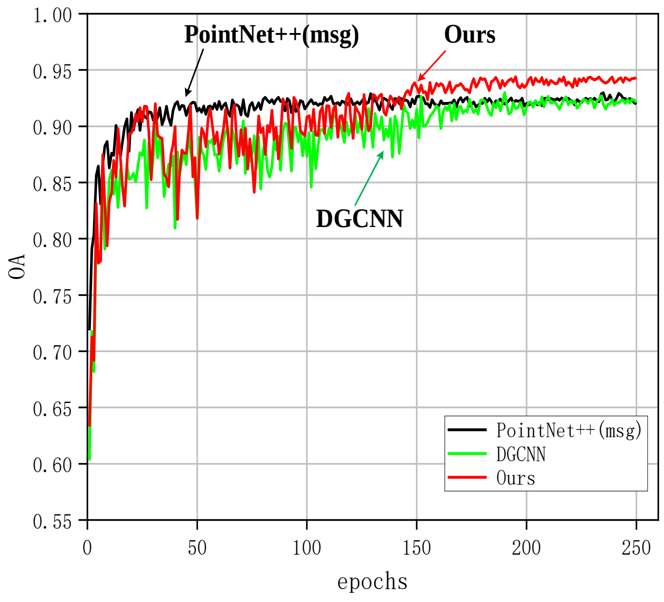

Fig. 5 Comparison diagram of *OA*

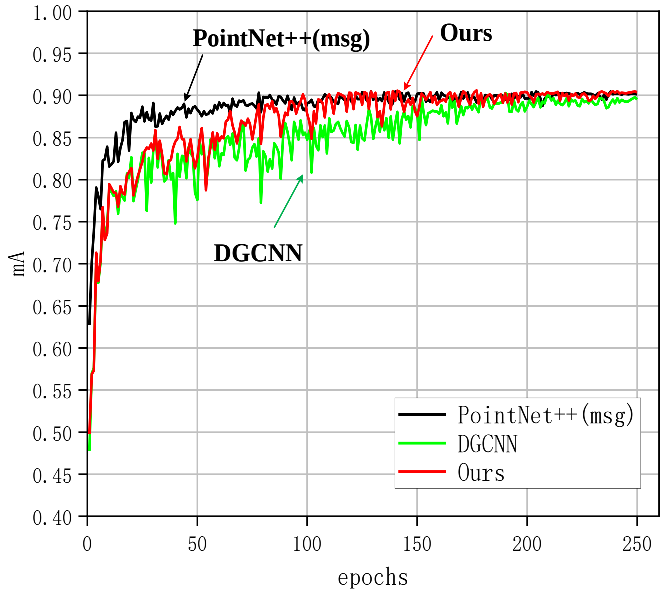

Fig. 6 Comparison diagram of *mA*

Also, in this thesis model, 40 categories in the dataset are classified and compared with PointNet++ and DGCNN, respectively. As shown in TABLE III, it verifies that this paper has improved overall classification accuracy for each category for the most part and proves that the model in this paper is more generalizable.

TABLE III
CATEGORY ACCURACY

Classification	PointNet++	DGCNN	Ours	Classification	PointNet++	DGCNN	Ours
Airplane	1.000	1.000	1.000	Laptop	1.000	1.000	1.000
Bathtub	0.861	0.922	0.941	Mantel	0.953	0.972	0.976
Bed	0.969	0.978	0.981	Monitor	0.961	0.981	0.985
Bench	0.700	0.762	0.846	Night stand	0.721	0.833	0.851
Book shell	0.913	0.925	0.942	Person	0.909	0.976	0.972
Bottle	0.945	0.972	0.976	Piano	0.860	0.956	0.961
Bow	0.903	0.954	0.962	Range hood	0.922	0.937	0.942
Car	0.986	0.990	0.991	Sink	0.804	0.807	0.814
Chair	0.973	0.981	0.987	Sofa	0.958	0.967	0.971
Cone	0.957	1.000	1.000	Stairs	0.858	0.954	0.964
Cup	0.802	0.786	0.843	Stool	0.855	0.878	0.885
Curtain	0.900	0.934	0.961	Table	0.819	0.956	0.963
Desk	0.802	0.943	0.951	Tent	0.952	0.961	0.968
Door	0.814	0.927	0.935	Toilet	0.980	0.960	0.971
Dresser	0.721	0.776	0.801	Television stand	0.797	0.89	0.883
Flower pot	0.208	0.705	0.763	Vase	0.812	0.814	0.821
Glass box	0.982	0.967	0.944	Wardrobe	0.612	0.795	0.814
Guitar	1.000	1.000	1.000	Xbox	0.856	0.848	0.855
Keyboard	1.000	1.000	1.000	Plant	0.762	0.743	0.788
Lamp	0.951	0.961	0.974	Radio	0.756	0.875	0.846

C. Model Analysis

i. Determine the Value of K and Radius

The core module of this paper is constructing a local domain using a combination of KNN and ball radius queries. Therefore, the experimental results of the model in this paper at $K = 5, 10, 15, 20$ and radius $r = 0.1, 0.2, 0.3, 0.4$ are shown in Figs. 7 and 8. During the model testing phase in this study, the optimal performance is observed when using a radius of 0.2 and a value of K equal to 20. As depicted in Figure 7, the *OA* metric achieved a peak performance of 93.8%. Similarly, Figure 8 illustrates the peak performance of the *mA* metric at 90.6%. Therefore, this paper sets the model's radius to 0.2 and the value of K to 20.

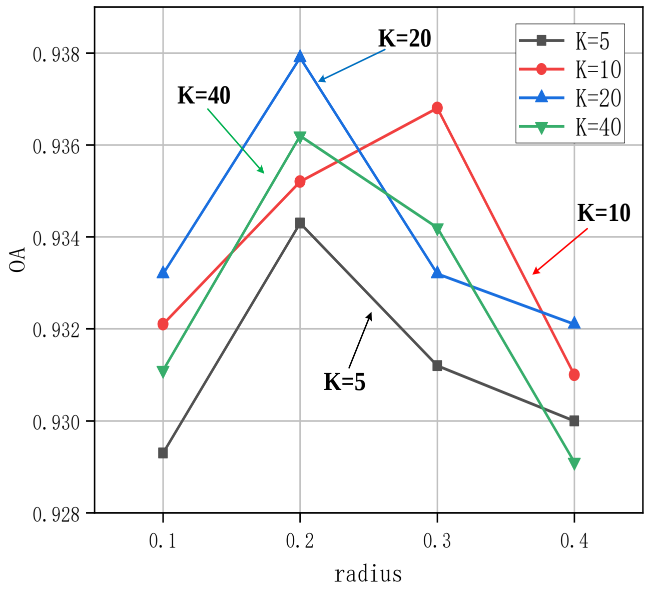

Fig. 7 Comparison diagram of *OA* of KNN-BR

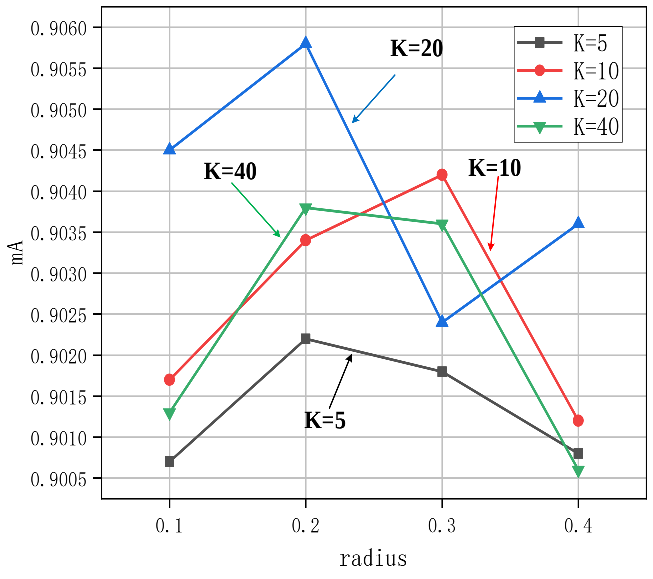

Fig. 8 Comparison diagram of *mA* of KNN-BR

ii. Determining the Pooling Method

To investigate the influence of different pooling methods on classification accuracy, the paper tests the models for maximum pooling, average pooling, and combining the two methods. Table IV presents the test results. Which √ indicates that the method is used and x indicates that the procedure is not used. The test results of maximum pooling and average pooling show that the maximum pooling results outperform the average pooling results. It is 0.15% higher in the *OA* metric and 0.11% higher in the *mA* metric, which is consistent with the conclusions obtained by PointNet. Combining the two pooling approaches improves the *OA* metric by 0.21% and the *mA* metric by 0.18% compared to using maximum pooling alone. This indicates that the constructed global pooling method can effectively reduce information loss in the international feature selection process.

Therefore, this article extracts local and global features from the model by building two pooling methods.

TABLE IV
GLOBAL POOLING TESTING

Max pooling	Ave pooling	OA	mA
√	x	93.59	90.42
x	√	93.47	90.31
√	√	93.8	90.6

V. CONCLUSION

This study uses DGCNN as the baseline model to improve three aspects of the point cloud classification task. First, the model proposed improves the KNN query by combining it with a spherical radius query to construct a neighborhood map. This enhancement aims to strengthen the local receptive field of features and enhance the characterization ability of point cloud features. Next, the paper adds a conventional convolutional layer to the model to expand its perceptual field and strengthen its abstraction. It is finally constructing a global pooling approach that combines maximum and average pooling to increase the focus on global features. The improved model achieved an accuracy of 93.8% on the ModelNet40 dataset, a significant improvement compared to the baseline model of 92.9%. The experiments show that the improved method proposed in this paper can significantly enhance the performance of the point cloud classification task.

Finally, we need to explore further the application effect of the point cloud classification method in practical applications. Although our model performs well in the point cloud classification task, its effectiveness and stability in practical applications need further validation. Therefore, we need to conduct more in-depth experiments and applications of the model and apply it to a broader range of fields and tasks. With the continuous development and advancement of point cloud technology, the research in this paper will make a more critical contribution to point cloud processing.

REFERENCES

[1] Shi, Shaoshuai, Xiaogang Wang, and Hongsheng Li, “Pointrcnn: 3D object proposal generation and detection from point cloud,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 770-779, 2019.

[2] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” *Proceedings of the International Conference on Machine Learning*, pp. 7354-7363, 2019.

[3] A. Johnson and M. Hebert, “Using spin images for efficient object recognition in cluttered 3D scenes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 433-449, 1999.

[4] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, “Recognizing objects in range data using regional point descriptors,” *Proceedings of the Computer Vision-ECCV 2004: 8th European Conference on Computer Vision*, Prague, Czech Republic, 11-14 May, 2004, Springer Berlin Heidelberg, pp. 224-237, 2004.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3D classification and segmentation,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 652-660, 2017.

[6] C. R. Qi, L. Yi, and H. Su, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *Advances in Neural Information Processing Systems*, pp. 5099-5108, 2017.

[7] 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 Transactions on Graphics (tog)*, vol. 38, no. 5, pp. 1-12, 2019.

[8] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” *Advances in Neural Information Processing Systems*, pp. 3844-3852, 2016.

[9] Z. Y. Deng, “Research and Application of Webpage Information Recognition Method Based on KNN Algorithm,” *IAENG International Journal of Applied Mathematics*, vol. 52, no. 3, pp725-731, 2022.

[10] H. Zhou, Y. Feng, M. Fang, M. Wei, J. Qin, and T. Lu, “Adaptive graph convolution for point cloud analysis,” *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4965-4974, 2021.

[11] Y. Zhou, and Q. Tuzel, “Voxelnet: End-to-end learning for point cloud based 3d object detection,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4490-4499, 2018.

[12] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, “Rotational projection statistics for 3D local surface description and object recognition,” *International Journal of Computer Vision*, pp. 63-86, 2013.

[13] W. Q. Qin, W. C. Zhao, and M. Li, “Multi-level Feature Representation and Multi-layered Fusion Contrast for Few-Shot Classification,” *IAENG International Journal of Computer Science*, vol. 49, no. 2, pp318-324, 2022.

[14] C. Z. Huang, and Y. Zhong, “A Network Representation Learning Method Fusing Multi-Dimensional Classification Information of Nodes,” *IAENG International Journal of Computer Science*, vol. 50, no. 1, pp94-105, 2023.

[15] Y. Zhang, M. Bai, P. Kohli, S. Izadi, and J. Xiao, “Deepcontext: Context-encoding neural pathways for 3D holistic scene understanding,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1192-1201, 2017.

[16] B. F. Ma, and Y. F. Chen, “Attentive Enhanced Convolutional Neural Network for Point Cloud Analysis,” *IAENG International Journal of Computer Science*, vol. 50, no. 2, pp417-421, 2023.

[17] S. R. S. Bhat, “A Counter Example for Neighbourhood Number Less Than Edge Covering Number of a Graph,” *IAENG International Journal of Applied Mathematics*, vol. 52, no. 2, pp500-506, 2022.

[18] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, “Multi-view convolutional neural networks for 3D shape recognition,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 945-953, 2015.

[19] D. Maturana, and S. Scherer, “Voxnet: A 3D convolutional neural network for real-time object recognition,” *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 922-928, 2015.

[20] H. Thomas, C. R. Qi, J. E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, “Kpconv: Flexible and deformable convolution for point clouds,” *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6411-6420, 2019.

[21] W. Shi, and R. Rajkumar, “Point-GNN: Graph neural network for 3D object detection in a point cloud,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1711-1719, 2020.

[22] W. Liu, J. Sun, W. Li, T. Hu, and P. Wang, “Deep learning on point clouds and its application: A survey,” *Sensors*, vol. 19, no. 19, pp. 4188-4210, 2019.

[23] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3D object reconstruction from a single image,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 605-613, 2017.

[24] X. Wang, S. Liu, X. Shen, C. Shen, and J. Jia, “Associatively segmenting instances and semantics in point clouds,” *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4096-4105, 2019.

[25] H. Zhou, Y. Feng, M. Fang, M. Wei, J. Qin, and T. Lu, “Adaptive graph convolution for point cloud analysis,” *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4965-4974, 2021.

[26] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3D ShapeNets: A deep representation for volumetric shapes,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1912-1920, 2015.