Point cloud classification by dynamic graph CNN with adaptive feature fusion

Rui Guo$^{1,2}$ | Yong Zhou$^{1,2}$ | Jiaqi Zhao$^{1,2}$ | Yiyun Man$^3$ | Minjie Liu$^{1,2}$ | Rui Yao$^{1,2}$ | Bing Liu$^{1,2}$

$^1$School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China
$^2$Engineering Research Center of Mine Digitization, Ministry of Education of the People's Republic of China, Xuzhou, China
$^3$Qian Xuesen Laboratory of Space Technology, Beijing, China

Correspondence
Yong Zhou, School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, China.
Email: yzhou@cumt.edu.cn

Funding information
National Natural Science Foundation of China, Grant/Award Number: 6180206; National Natural Science Foundation of China, Grant/Award Number: 61772530; Six Talent Peaks Project in Jiangsu Province, Grant/Award Number: 2015-DZXX-010; Natural Science Foundation of Jiangsu Province, Grant/Award Number: BK20100359; Natural Science Foundation of Jiangsu Province (China), Grant/Award Number: BK20171192; China Postdoctoral Science Foundation, Grant/Award Number: 2018M642359

Abstract
The deep neural network has made the most advanced breakthrough in almost all 2D image tasks, so we consider the application of deep learning in 3D images. Point cloud data, as the most basic and important form of representation of 3D images, can accurately and intuitively show the real world. The authors propose a new network based on feature fusion to improve the point cloud classification and segmentation tasks. Our network mainly consists of three parts: global feature extractor, local feature extractor and adaptive feature fusion module. A multi-scale transformation network is devised to guarantee the invariance of the transformation of the global feature, and a residual block is introduced to alleviate the problem of gradient disappearance to enhance the global feature extractor. Based on the edge convolution and multi-layer perceptron, a local feature extractor is constructed. Finally, an adaptive feature-fusion module is proposed to complete the fusion of global features and local features. Extensive experiments on point cloud classification and segmentation tasks are carried out to verify the effectiveness of the proposed method. The classification accuracy of the ModelNet40 is 93.6%, which is 4.4% higher than that of the PointNet. Similarly, the segmentation accuracy on the ShapeNet is 85.6%, which is higher than other methods.

1 | INTRODUCTION

The point cloud is a collection of data that expresses the distribution of spatial objects and the characteristics of the surface of objects in the same spatial reference frame [1]. Point cloud data can be obtained directly from the Lidar scanners, Radars and depth cameras. After getting the spatial coordinates of each sampling point on the object surface, the collection of points obtained is named point cloud. As a form of image representation, the point cloud is different from the 2D image data due to the lack of depth information. The availability of inexpensive 3D sensors has made point cloud data widely available; besides, the point cloud data can keep the absolute scale and position of the object and is more insusceptible to the external factors. Specifically, point cloud data refer to the combination of a set of vectors in a 3D coordinate system, usually in the form of 3D coordinates (x, y, z), generally used to represent the surface shape of an object. Point cloud data can represent not only the most basic 3D geometric information, but also the colour information, grey value and so on. Point cloud data have wide application value in 3D reconstruction [2], point cloud registration, self-driving [3], indoor automatic navigation [4] and robotics [5]. Therefore, point cloud learning is particularly meaningful and we mainly focus on the two classic tasks: point cloud classification and point cloud segmentation.

Point cloud classification divides the input point set into specific different categories, and there are altogether 40 categories in the modelnet40 dataset. The point cloud...
segmentation task outputs the label of each point. We tested our methods on two datasets, ModelNet40 [6] and ShapeNet [7], respectively. It can be seen from the description of the point cloud that it is obviously different from the 2D image [8] so that the classical model such as CNN cannot be directly used. How to process the point cloud data reasonably and accurately is an important problem. It is a natural approach to convert the 3D point cloud data into a 2D image [6,9,10]. However, this method has no access to the segmentation task. Another solution transforms the point cloud data into a structured 3D grid, but voxels can be pretty expensive in computing [11,12]. In recent years, the mature application of deep learning on 2D images has provided a new idea. How to apply the convolutional neural network in the field of 2D images to point cloud data processing is a problem.

Here, we propose a new point cloud processing scheme that directly processes the point cloud. Our network is divided into three main parts: global feature extractor, local feature extractor and adaptive fusion module. We used PointNet as the baseline of the global feature extractor. The authors of PointNet designed a point cloud transformation net named T-Net, which is mainly used to eliminate the impact of rigid point cloud transformation on the overall network and ensure the rotation of the point cloud without deformation. However, this scheme only considers the characteristics of the highest dimension. We devise a multi-scale transformation net to ensure the rotation invariance of point cloud while ensuring the integrity of input information, which extract multi-layer features from the input by splicing the low-level layer feature, middle-level features and high-level features. Also, we add a residual block to alleviate the gradient disappearance problem. With these two improvements, we strengthen the global feature extractor. Then, we use the edge convolution (edgeconv) and multi-layer perceptron (MLP) to construct the local feature extractor and we finally design an adaptive feature fusion module to allocate the weight of the local feature and global feature adaptively to complete the fusion of global and local feature.

Our main contributions are as follows:

- We devise a multi-scale transform net to improve the alignment of point clouds, which extracts features from the input point cloud and its multi-layer features.
- We adapt residual connections to mitigate the gradient disappearance in the global feature extractor.
- We devise a new adaptive feature fusion method to complete the fusion of local features and global features.
- We show how to build our network to achieve the improved result on ModelNet40 for classification tasks and evaluate our network on ShapeNet for segmentation tasks.

2 | RELATED WORKS

Deep learning is bringing change to many areas and new ideas to many pending directions. For example, pattern recognition, computer vision and data analysis have developed rapidly after the application of deep learning [14]. In the field of 2D image, deep learning has shown a strong driving force. Deep learning can always be seen in image classification [15,16], medical image processing [17,18], object detection [19–21] and semantic segmentation [22,23]. Inspired by the multi-view-based method proposed by [24], and the voxel-based idea proposed by [11], deep learning becomes popular in the field of 3D images. Herein, we will discuss some significant approaches used in point cloud learning. Specifically, we focus on the following methods: multi-view-based methods, volume-based methods, PointNet-based methods and graph-based methods.

2.1 | View-based methods

The speed and accuracy of the 2D image processing are much higher than those of the 3D image processing in the year 2015. As a result, some researchers investigated the methods for converting the 3D data into 2D data; thus, the typical CNN architecture is available again. Multi-view Convolutional Neural Networks for 3D Shape Recognition (MVCNN) [24] collects a series of 2D images of the same 3D object from multi perspectives. Then, CNN is applied for each 2D image to extract key features, and the features obtained from different perspectives are aggregated by the view-pooling layer at last. This work dramatically outperforms what was available at that time. Group-View Convolutional Neural Networks for 3D Shape Recognition (GVCNN) [25] is inspired by MVCNN [24], they take the differences between different perspectives into account, they divide the features with high similarity into the same group and assign the same influence factors which allow key feature weights to be adjusted to make a greater contribution to the whole work. It is a pity that these view-based approaches have no regard for some important drawbacks. Although a large number of 2D images can be used to approximate the real 3D scene, there is an inevitable loss of information in geometry such as the lack of depth information; this defect is fatal when considering semantic segmentation. Besides, it is possible to surround small objects with multiple views, while it is almost impossible to achieve a full coverage for complex scenes or large objects.

2.2 | Volume-based methods

In the field of 3D images, voxelization is to transform the geometric representation of an object into the voxelization representation that is closest to the object and generate volume data set, which not only contains the surface information of the model but also describes the internal attributes of the model. The spatial voxel representing the 3D model is similar to the 2D pixel representing the image, except that it expands from the 2D point to the 3D cube element. In short, voxelization is a reasonable solution to solve both unordered and unorganized problems of the raw point cloud. VoxNet [11] is the most famous voxel-based method which converts unorganized point cloud data into a 3D regular grid, then there is 3D CNN to predict the types of objects. Nevertheless, this method can lead to high storage and computing costs for the reason that it stores
not only the spaces already in use, but also free and uncertain data. Besides, 3D voxel always has pool resolution and it is unrealistic for the large-scale point cloud. Owing to the sparsity of input data, a series of unbalanced octree is used to divide the space to reduce unnecessary computation and memory consumption [26,27]. Each leaf node in the octree stores a pooled feature representation. This work focuses on memory allocation and computation in the relevant dense areas and enables deeper networks to handle a higher resolution.

2.3 | PointNet-based methods

The popularity of the point cloud cannot be separated from a landmark work PointNet [1], which pioneers the deep learning method to process irregular point cloud data. Different from the previous indirect use of point cloud data, PointNet uses the MLP to extract features directly from each point with shared parameters and there is no convolution operator in PointNet. Figures 1 and 2 describe the disordered characteristic of the point cloud. If we transform the point cloud into the image or volume grid, it is inevitable to consider the application scheme of convolution in regular space for the unorganized point cloud. Therefore, PointNet directly processes point cloud data without mediation tools to bypass the structural limitations of the point cloud. Point cloud data are an irregular set of points, and the convolution applied to the point cloud should have permutation invariance to the input order of the point cloud. To meet this requirement, PointNet introduces an MLP to extract features from each input point separately, MLP faces different inputs with the same shared parameters and this feature can extract the same category of features from different points. As for the symmetry invariance of point cloud, the core of PointNet is that MLP is applied to aggregate the extracted local features into global features while guaranteeing the invariance of replacement and extracting features by a symmetric max-pooling maximization. Both MLP and max-pooling functions with shared parameters have the characteristics of symmetry invariance; as a result, the symmetry invariance of the point cloud is guaranteed. In addition, the transformation network in the PointNet network maintains the transformation invariance of the point cloud data based on a series of matrix operations.

The work of PointNet is pioneering; however, PointNet lacks local information and the extraction of feature of points alone does not pay attention to the relationship between the adjacent points. The connection between the points is more robust in the transformation of the point cloud than the information provided by the independent points. Therefore, the authors of the PointNet put forward the PointNet++ [28]. PointNet++ is devised to compensate for the lack of local characteristics to the attention of the PointNet. PointNet++ uses the idea of hierarchical feature extraction and the farthest point sampling method to collect some relatively important points as the centre point from the point cloud. In a certain range of the centre point, it uses KNN to get some neighbouring points to generate the patch and uses PointNet to extract the features of these patches as the features of the centre point. Then, the next layer is sent to continue the same operation so that the centre points of each layer are a subset of the centre points of the previous layer, and with the deepening of the number of layers, the number of centre points is less and as the number of layers deepens, the number of center points decreases, but each centre point contains more and more information. Although PointNet++ enhances the accuracy of the point cloud processing, it also does not take the relationship between the points into consideration. Based on the idea of stratification in PointNet++, the sift module in the 2D image is introduced into the 3D point cloud in PointSIFT [29], and features in various directions are extracted mainly through directional coding and scale perception to improve the description of the local features.

As opposed to the convolution-free structure PointNet-based model, a growing number of researchers are turning to PointNet alternatives, considering the application of the convolution to the point cloud. SpiderCNN [30] states that MLP is not a very nice way to extract features for point cloud; thus, they design a new network structure based on the SpiderCNN layer. SpiderCNN extracts local geodesic information by the Taylor polynomial. It is not an easy idea in which a lot of complicated mathematics is used. However, the work does not exceed the highest accuracy of the network based on MLP. PointCNN [31] uses an X-transformation instead of symmetric functions to normalize the order, which is the promotion of CNN in the field of point cloud. PointConv [32] is a density-reweighted convolution that completely approximates a 3D continuous convolution at any set of 3D point cloud data. PointConv is implemented by introducing an effective method to improve the storage efficiency. Most importantly, this method can achieve the same translation invariance as the 2D convolution network and the invariance of the order in the point cloud. SPLATNet [33] provided a framework for merging 2D images with 3D point clouds and preserves spatial information even in sparse regions. Bilateral convolution layers (BCLs) which contain three basic operations—Splat, Convolve and Slice— that are used to construct the SPLATNet network structure. The point cloud is put into every convolution
operation to realize end-to-end processing. RS-Net [34] put forward a slice pooling layer; points are firstly sliced in each direction and then points are evenly divided into N independent parts in each direction. Features are aggregated by max-pooling for each part respectively so that the feature vectors obtained are orderly and structured.

2.4 | Graph-based methods

The researchers also tried to improve the PointNet to allow it to consider the learning characteristics of relationships between the graph-based points and the adjacent points. ECC [35] is the first work to use the graph in point cloud learning, and it describes the specific method of composition on the original point cloud data structure as:

- Let us view the point cloud as a graph structure, where each point in the point cloud is viewed as a vertex in the graph structure.
- The initial value of each vertex is set to the characteristics of the corresponding point, and the spatial neighbourhood is generally obtained by KNN.
- All the points in the spatial neighbourhood of the vertex are considered as the neighbours of the vertex.
- Use directed edges to connect vertices to their neighbours to complete a point cloud composition.

DGCNN [13] devises an edge convolutional operator to extract the feature from a centre point and the edge feature from its neighbour; this work connects the graph to point cloud and inspires many researchers to pay attention to the graph CNN. GAP-Net [36] and AGCN [37] make an effort to apply the attention mechanism which focuses on the local feature to the graph neural network, there is a slight improvement in the classification accuracy than that of dynamic graph CNN. LDGCNN [38] is heavily inspired by DGCNN [13]. The network performance has been improved by adding a jump connection structure[39]. The following work is DGCNN [13]. Some improvements have been made to their previous work, including replacing fixed graphs with dynamic ones and reconstructing the diagram after each edgeconv layer changes the ReLU activation function to Leaky ReLU.

3 | METHODS

3.1 | Problem statement

Our method focuses on the point cloud data. Generally, point cloud appears as a set of \( n \) points.

\[
P = \{p_1, \ldots, p_n\} \subseteq \mathbb{R}^3
\]  

(1)

If we only need three dimensions for each point, \( p \) is defined as Equation (2).

\[
p = \{x_i, y_i, z_i\}, p \subseteq \mathbb{R}^3
\]  

(2)

Surely, our set can be \( \mathbb{R}^2 \), \( \mathbb{R}^3 \), \( \mathbb{R}^6 \) and so on, then the input of our network includes colour, surface normal and so on. The \( c \) in our network also represents the channel in the layer. The point cloud is a set of points without a specific order. So we implement whatever the order of the point cloud is we also extract the same feature, the same as PointNet [1], we use a simple symmetric function max-pooling to achieve this goal. No matter what the order is, we also choose the same maximum of the feature in all the channels.

3.2 | Global feature extractor

Our global feature extractor is inspired by the PointNet [1]. Experiments prove that the T-Net which is used to align point cloud in PointNet brings significant improvement to network performance. T-Net maps the input feature to high-dimension space by MLP and then reshapes the high-dimension feature to the same dimension with the input feature by the matrix transformation and a series of full connection layers. This scheme only focuses on the highest dimension feature information, which may lead to the loss of the original information. As shown in Figure 3, we spliced the input data, low-level features and middle-level features into high-dimensional
features using skip links, enhancing the ability of the network to extract semantic and geometric information. As we all know, the optimization of the neural network is optimized by the gradient descent. However, vanishing gradients is a common phenomenon in deep learning. The original information is transmitted to the subsequent convolution layer, which has a positive effect on the compensation of the original information, so we apply the residual module in the global feature extractor by adding the direct link between the input and the convolution layer. The dimension of $A_0 \times A_1$ is respectively $n \times 64$, $n \times 128$, $n \times 1024$, $n \times 512$, $n \times 256$. When MT-Net is used for input transform, the dimensions of input and output are both $n \times 3$. When MT-Net is used for feature transform, the dimensions of input and output are both $n \times 64$.

### 3.3 Local feature extractor

As we discussed in Section 2, the application of graphs in the point cloud processing provides a solution for learning about the relationship between a point and its neighbourhood. It is a better method to describe the local features of the point cloud.

In the local feature extractor, we compute the graph to represent the local point cloud structure:

$$ G = (V, E), V = \{p_1, p_2, \ldots, p_n\}, E \subseteq V \times V $$

(3)

When we focus on a picked point, we need to use the appropriate method to choose the key point as its neighbours instead of directly picking all the points. As the Figure 4 shows, we put the KNN into use. To be specific, for the central node $p_i$, edges $e_i$ represent the relationship between central node and its neighbours which are chosen by KNN. $p_{j}$ is one neighbour of point $p_i$ and $e_j$ represents the directed edge from point $p_j$ to $p_i$. The edge feature $e_j$ is calculated as:

$$ G = (V, E) $$

$$ V = \{p_1, p_2, \ldots, p_n\} $$

$$ E = \{e_1, e_2, \ldots, e_k\} $$

$$ e_i = \{e_{i1}, e_{i2}, \ldots, e_{ik}\} $$

$$ e_j = p_j - p_i $$

(4)

Now, we complete the construction of the local graph. Next, we extract the local features of the graph we construct based on the edge convolution. We use the same feature extraction function for all points, so we select an arbitrary point as the centre point to illustrate how to use the centre point and its k-nearest neighbour to extract local features. The input of our network is the local graph of the central point $p_i$ and the output of edgeconv is the feature $l_i$.

$$ l_i = \max\{h_0(p_i; e_{i1}), h_0(p_i; e_{i2}), \ldots, h_0(p_i; e_{ik})\} $$

(5)

where $h_0$ represents a linear function with a set of learnable parameters. The maximum pooling function is used to aggregate features, because it is independent of the arrangement of points and can extract the most important features of all edges.
3.4 | Feature fusion

In our network, the local feature extractor uses KNN to search the adjacent points of each central point and uses the central point and the adjacent point to construct the graph. Edge convolution describes the relationship between points, accurately extracts local features of point clouds and takes point-to-point contact information as the edge information in the graph. However, this method cannot describe the information of the point itself exactly, so we introduce the global feature extractor to supplement the global feature through the feature fusion. The global feature extractor uses the improved PointNet of residual blocks to extract features from each point in the input point cloud and aggregate the feature information of each point into global features. In addition, we use MT-Net instead of T-Net. In MT-Net, input data, low-level features and middle-level features are spliced into high-dimensional features through skip links, which improves the module’s ability to extract semantic and geometric information. In deep learning, it is an important way to integrate the features of different scales to improve the network performance [40]. The features that are low level have higher resolution and lower semantics, while the features that are high level have stronger semantic information and lower resolution. The key to improving the classification and segmentation is to fuse the multi-scale features efficiently, so we apply the feature fusion to point the cloud classification task. Vector stitching is a classical feature fusion method, which loses sight of the importance of different features. Different features have different effects on the network; hence, we devise a new adaptive feature fusion method for point cloud classification and segmentation tasks. For global features and multi-scale local features, characteristics of different scales may have different contributions to classification results, so we assign an initial influence weight to them respectively. Then through training to adjust the weight, we finally assign larger weights to key features and smaller weights to those relatively unimportant features. Our idea is simple and effective. The details are as follows.

We get a global feature $n \times 128$ by the global feature extractor, and in the local feature extractor, we focus on these features after the edgeconv layer, there are four local features with multi-channel. To achieve the multi-scale feature fusion, we map features to the same channel $n \times 128$ adapted to the global feature. We use $F, L_1, L_2, L_3$ respectively to express global feature and features in the local feature extractor. Then, we initialize an array of weights through the softmax function to map the parameters to a range $(0, 1)$. The weight array $x$ is expressed as Equation (6).

$$x = [x_0, x_1, x_2, x_3, x_4]$$ (6)

The feature $P$ represents the characteristics of the entire point cloud input which is calculated as:

$$P = x_0 * F + x_1 * L_1 + x_2 * L_2 + x_3 * L_3 + x_4 * L_4$$ (7)

The parameters $x_0, x_1, x_2, x_3, x_4$ are trainable.

3.5 | Architecture

We adapt our models for classification tasks and segmentation tasks, as shown in Figures 5 and 6 respectively. For classification tasks, we use input transform to make the input points to geometric transformations, use feature transform to adjust the feature in the global feature extractor and implement the residual module with skip collection. In the local feature extractor, the core idea of the edgeconv is to pay attention to the connection between the point pairs determined by KNN. Then, multi-scale features are adaptively combined. Next, we use the shared full-concerned layer and a max-pooling layer to obtain a feature for entire input points. Finally, the MLP layers (512, 256, 40) and the dropout operation are used to get the predicted score of 40 categories. The segmentation framework is modified from our classification network which directly concatenates the global feature and local feature, then the categorical vector is linked to the aggregated feature by

---

**Figure 6** The segmentation framework is modified from our classification network which directly concatenates the global feature and local feature, then the categorical vector is linked to the aggregated feature by max-pooling layer. The classification scores of per-point for $p$ semantic labels are obtained by MLP layers (512, 256, 128, $p$).
max-pooling layer. The classification scores of per-point for semantic labels are obtained by MLP layers (512, 256, 128, p) (Fig. 6).

4 | EXPERIMENTS AND DISCUSSIONS

4.1 | Data description

Our work is mainly verified on two datasets, one is the ModelNet40 [6] for point cloud classification tasks, which contains 12, 311 CAD models of 40 different categories; this dataset is split into a training set (9843 models) and a validation set (2468 models) [1]; we just use the 3D coordinates (x, y, z) of the point. The other is the ShapeNet part dataset [7] for segmentation tasks, which is a set of 16, 881 CAD models from 16 categories. Each CAD model was sampled to 2048 points by former scholars, each of which is annotated with one of 50 parts. Researchers sample each CAD model to 2048 points.

4.2 | Experimental environment

The proposed method for classification tasks is implemented using the Python language and the Pytorch library, which is a deep learning tensor library using GPU and CPU optimizations. The proposed method for the segmentation task is implemented using Python language and the TensorFlow library. All the experiments are trained on a graphics workstation (6 CPUs E5-2609v4@1.70GHz, 16-GB memory, and 1 P100 GPUs).

4.3 | Classification

We have a description for our training data in Section 4.1. The implementation details of our experiment are the same as [13]. The SGD optimizer with a learning rate of 0.1 is used; we apply the cosine annealing [41] to reduce the learning rate until 0.001. We set the momentum at 0.9 for batch normalization and discard the batch normalization decay. Besides, the batch size is 32.

We evaluate our model on the ModelNet40 dataset [6] to test the classification result. We compare our method with some excellent related work and record it in Table 1. We retrained the classification accuracy of PointNet++ and SpiderCNN to obtain the average classification accuracy, so it is slightly different from the classification accuracy in the original paper. As Table 1 shows, our network has a visible improvement on the classification accuracy for the same input point cloud; we improve the classification result from 92.9% to 93.6%.

We have verified the effectiveness of the multi-scale transform net in our network; besides, the embeddability of our module is proved by the experiment on PointNet. We only changed the module of feature transformation in

| Methods | Mean class accuracy (%) | Overall accuracy (%) |
|---------|-------------------------|----------------------|
| 3DShapeNets [6] | 77.3 | 84.7 |
| VoxNet [11] | 83.0 | 85.9 |
| ECC [35] | 83.2 | 87.4 |
| PointNet [1] | 86.0 | 89.2 |
| PointNet++ [28] | 87.5 | 89.9 |
| SpiderCNN [30] | 89.0 | 91.8 |
| KD-NET [42] | 88.5 | 91.8 |
| PointCNN [31] | 88.1 | 92.2 |
| DGCNN [13] | 90.2 | 92.9 |
| LDGCNN [38] | 90.3 | 92.9 |
| Ours | 90.6 | 93.6 |

| Methods | Mean class accuracy (%) | Overall accuracy (%) |
|---------|-------------------------|----------------------|
| PointNet (official) | 89.2 | |
| T-Net | 90.2 | 90.7 |
| MT-Net | 86.2 | 86.5 | 87.0 |

PointNet without any other changes to the network, and the experimental results were significantly improved as shown in Table 2.

In addition, we devise a group of experiments to prove the effectiveness of each module. As shown in Table 3, each module contributes to the accuracy of the classification. ‘Residual + MT-Net’ means the residual module added in global feature extractor and replace the T-Net with our MT-Net, ‘Adaptive Fusion’ means the adaptive-fusion module. Both modules work individually with slight improvements, and there is about 0.7% improvement when both modules are applied at the same time. Furthermore, we also pay attention to the influence of the different numbers \(k\) which is the parameter of the nearest neighbours. In this test, we just adjust the parameter \(k\) without other changes. As Table 4 shows, the limit experiments suggest we set \(k\) to six in our final model. The result also indicates that compared with DGCNN, our network performs well under different \(k\), which reflects the robustness of the network. Also, we analyse the model complexity, as shown in Table 5. Although our method is implemented based on feature fusion, the model size and time complexity are not significantly improved compared with previous work. While ensuring the high efficiency of the network, the lightweight of the network is also taken into account.
4.4 | Segmentation

The Intersection-over-Union (IOU) is a measure of the accuracy of an object in a particular data set. For the segmentation task on the ShapeNet, we use IOU to make a quantitative description of our network, and our segmentation model is shown in Figure 6. The segmentation network predicts the label of each point, and we compare the predictions with the ground truth. Then the intersection and union of predicted points are calculated. The IOU is the ratio of the number of points in the intersection to that in the union. The segmentation method was modified from our class network. As shown in Table 6, we compare our results with several related works. We do perform best in overall categories, and we achieve the best scores in seven collections of the ShapeNet dataset. Besides, we make visualizations of segmentation results to get a feel of just how well the model is performing in Figure 7.

5 | DISCUSSION

Herein, we propose an adaptive feature-fusion-based point neural network to improve the point cloud processing. We design an adaptive feature-fusion module that compensates for the relative information of individual points in the local feature extractor by global feature extractor. Through experiments, we get improved performance on the ModelNet40 dataset and provide a method for feature fusion to complete the point cloud segmentation tasks on the ShapeNet. Also, we use data visualization to qualitatively demonstrate the point cloud segmentation results. Our work shows that the classical methods used in 2D images are equally applicable in point cloud when we find the right way.

Some of the previous work of direct point cloud processing, such as PointNet, only describes the global characteristics and does not take into account the local characteristic of the point cloud. PointNet++ uses the idea of layering to process the point cloud data, and uses PointNet to extract the features of each layer, repeat this hierarchical feature extraction operation, and finally use feature aggregation to obtain the output of global and local information, but PointNet++ does not consider the connection between points. DGCNN constructs the graph to complete point cloud learning and the relationship between points can be represented by the edge in the graph, so as to achieve the description of the contact between points. However, DGCNN does not deal with the global features delicately. Our network considers the complete point cloud with the graph structure local feature extractor and the local information extraction, using the MT-Net and residual block, to enhance PointNet as our global feature extractor, to design an adaptive feature fusion module for assigning appropriate weights global features and local characteristics to guide the fusion of the global and local feature, and effectively improve the accuracy of the point cloud processing. The disadvantage is that when we use the graph structure, we may not be able to learn the difference of point-to-point connections well. Besides, the application of the feature fusion inevitably increases the model size and time complexity, although these defects should be acceptable when considering the performance improvement.

In the future, we will try to introduce the attention mechanism into the graph neural network to better learn the relationship between points and make an attempt to other loss function such as the focal loss to replace the exiting loss function; besides, the performance of recent hypergraphs in the point cloud learning is excellent so that we will try to replace the graph structure with a hypergraph; we will further improve the feature fusion module for segmentation tasks and apply our approach to more point cloud datasets and combine it with the practical application.
TABLE 6 Part segmentation results on the ShapeNet part dataset, we calculate the ratio of the intersection and union of the true value and the predicted value to get the mIOU value as our evaluation indicator. For the values with the highest accuracy, we have a bold highlighting.

|        | Mean | are. | bag | cap | Car | cha. | ear. | gui. | kni. | lam. | lap. | mot. | Mug | pis. | roc. | ska. | tab. |
|--------|------|------|-----|-----|-----|------|------|------|------|------|------|------|-----|------|------|------|-----|
| Dgcnn  | 86.1 | 85.7 | 85.4 | 85.3 | 85.2 | 85.1 | 85.0 | 84.9 | 84.7 | 84.5 | 84.3 | 84.1 | 84.0 | 83.9 | 83.8 | 83.7 | 83.6 |
| P # Shapes | 5271 | 535 | 543 | 541 | 539 | 537 | 535 | 533 | 531 | 529 | 527 | 525 | 523 | 521 | 519 | 517 | 515 |
| Dgcnn++ | 85.1 | 85.0 | 84.9 | 84.8 | 84.7 | 84.6 | 84.5 | 84.4 | 84.3 | 84.2 | 84.1 | 84.0 | 83.9 | 83.8 | 83.7 | 83.6 | 83.5 |
| Lfnet  | 84.3 | 84.2 | 84.1 | 84.0 | 83.9 | 83.8 | 83.7 | 83.6 | 83.5 | 83.4 | 83.3 | 83.2 | 83.1 | 83.0 | 82.9 | 82.8 | 82.7 |
| kd-net | 82.3 | 82.2 | 82.1 | 82.0 | 81.9 | 81.8 | 81.7 | 81.6 | 81.5 | 81.4 | 81.3 | 81.2 | 81.1 | 81.0 | 80.9 | 80.8 | 80.7 |
| Dgenn  | 85.1 | 84.2 | 83.7 | 84.4 | 77.1 | 90.9 | 78.5 | 91.5 | 87.3 | 82.9 | 96.0 | 67.8 | 93.3 | 82.6 | 59.7 | 75.5 | 82.0 |
| Ours   | 85.6 | 85.3 | 81.0 | 85.2 | 79.6 | 86.1 | 75.7 | 92.2 | 86.4 | 81.2 | 95.9 | 87.7 | 74.8 | 68.2 | 59.5 | 75.1 | 83.2 |

FIGURE 7 We showed qualitative results of some of the objects for the part segmentation. We give different colours to different parts of an object. The top line is a visual representation of the input point cloud, and the bottom line is a visual representation of the part segmentation results. Since there is no obvious gap between the visual presentation of various methods, we only compare the difference between the results of the partial segmentation of our schemes and the input data.

ACKNOWLEDGEMENT
This work was partially supported by the National Natural Science Foundation of China (No. 61,806,206, 61,772,530), the Natural Science Foundation of Jiangsu Province (No. BK20180639, BK20201346, BK20171192), National Defence Basic Scientific Research Project (No. JCKY2017203C108), the China Postdoctoral Science Foundation (No. 2018M642359), and the Six Talent Peaks Project in Jiangsu Province (No. 2015-DZXX-010).

REFERENCES
1. Qi, C.R., et al.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Hawaii, pp. 652–660 (2017).
2. Qi, C.R., et al.: Frustum pointnets for 3d object detection from rgb-d data. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, pp. 918–927 (2018).
3. Ku, J., et al.: Joint 3d proposal generation and object detection from view aggregation. In: 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), Madrid, pp. 1–8 (IEEE 2018).
4. Zha, Y., et al.: Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: 2017 IEEE international conference on robotics and automation (ICRA), Marina Bay Sands, pp. 3357–3364 (2017).
5. Zhang, K., et al.: Environmental features recognition for lower limb prostheses towards predictive walking. IEEE Trans. Neural Syst. Rehabil. Eng. 27(3), 465–476 (2019).
6. Wu, Z., et al.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Boston, pp. 1912–1920 (2015).
7. Yi, L., et al.: A scalable active framework for region annotation in 3d shape collections. ACM Trans. Graph. 35(6), 1–12 (2016).
8. Girshick, R., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Columbus, pp. 580–587 (2014).
9. Yavartanoo, M., Kim, E.Y., Lee, K.M.: Spenet: Deep 3d object classification and retrieval using stereoscopic projection. In: Asian conference on computer vision, pp. 691–706 (Springer, Berlin 2018).
10. Chen, S., et al.: Veram: View-enhanced recurrent attention model for 3d shape classification. IEEE Trans. Visual. Comput. Graph. 25(12), 3244–3257 (2018).
11. Muratana, D., Scherer, S.: Voxnet: a 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS), Hamburg, pp. 922–928 (2015).
12. Wang, Z., Lu, F.: Vossegnet: Volumetric CNNs for semantic part segmentation of 3d shapes. IEEE Trans. Visual Comput. Graph. (2019).
13. Wang, Y., et al.: Dynamic graph CNN for learning on point clouds. ACM Trans. Graph. 38(5), 1–12 (2019).
14. Zhu, X.X., et al.: Deep learning in remote sensing: a comprehensive review and list of resources. IEEE Geosci. Remote Sens. Mag. 5(4), 8–36 (2017)

15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, Lake Tahoe, pp. 1097–1105 (2012)

16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). arXiv preprint arXiv:1409.1556

17. Öztürk, Ş., Akdemir, B.: Hic-net: A deep convolutional neural network model for classification of histopathological breast images. Comput. Electr. Eng. 76, 299–310 (2019)

18. Simonyan, K., Zisserman, A.: Very deep convolutional networks for 3d shape analysis. ACM Trans. Graph. 36(4), 1–11 (2017)

19. Chen, L., et al.: PointCNN: Convolution on x-transformed points. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2019). arXiv preprint arXiv:1905.13445

20. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, pp. 770–778 (2016)

21. He, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, pp. 770–778 (2016)

22. Zhang, K., et al.: Linked dynamic graph CNN: Learning on point cloud via linking hierarchical features (2019). arXiv preprint arXiv:1906.10014

23. Li, M., et al.: Multi-modal feature fusion for geographic image annotation. Pattern Recogn. 73, 1–14 (2018)

24. Guo, R., Zhou, Y., Zhao, J., et al.: Point cloud classification by dynamic graph CNN with adaptive feature fusion. IET Comput. Vis. 2021;15:235–244. https://doi.org/10.1049/cvi2.12039

25. Su, H., et al.: Multi-modal feature fusion for geographic image annotation. Pattern Recogn. 73, 1–14 (2018)

26. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts (2016). arXiv preprint arXiv:1608.03983

27. Redmon, J., et al.: You only look once: Unified, real-time object detection. In: Computer vision & pattern recognition, Las Vegas (2016)

28. Wang, P.-S., et al.: O-CNN: Octree-based convolutional neural networks for 3d shape analysis. ACM Trans. Graph. 36(4), 1–11 (2017)

29. Qi, C.R., et al.: PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems, Long Beach, pp. 5099–5108 (2017)

30. Xu, Y., et al.: Spidercnn: Deep learning on point sets with parameterised convolutional filters. In: Proceedings of the European conference on computer vision (ECCV), Munich, pp. 87–102 (2018)

31. Li, Y., et al.: PointCNN: Convolution on x-transformed points. In: Advances in neural information processing systems, Montréal, pp. 820–830 (2018)

32. Wu, W., Qi, Z., Fuxin, L.: PointConv: Deep convolutional networks on 3d point clouds. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Long Beach, pp. 9621–9630 (2019)

33. Su, H., et al.: Splatnet: sparse lattice networks for point cloud processing. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, pp. 2530–2539 (2018)

34. Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3d segmentation of point clouds. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, pp. 2626–2635 (2018)

35. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Hawaii, pp. 3693–3702 (2017)

36. Chen, C., Fragnonara, L. Z., Tsourdos, A.: GapNet: graph attention based point neural network for exploiting local feature of point cloud (2019). arXiv preprint arXiv:1905.08705

37. Xie, Z., Chen, J., Peng, B.: Point clouds learning with attention-based graph convolution networks (2019). arXiv preprint arXiv:1905.13445

38. Zhang, K., et al.: Linked dynamic graph CNN: Learning on point cloud via linking hierarchical features (2019). arXiv preprint arXiv:1904.10014

39. Li, K., et al.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Las Vegas, pp. 770–778 (2016)

40. Loshchilov, I., Hutter, F.: SGDR: Stochastic gradient descent with warm restarts (2016). arXiv preprint arXiv:1608.03983

41. Klokov, R., Lempitsky, V.: Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: Proceedings of the IEEE international conference on computer vision, Venice, pp. 863–872 (2017)