




PointFormer: A Dual Perception Attention-Based Network for Point Cloud Classification

Yijun Chen^{1,2,3}, Zhulun Yang^{1,2,3} , Xianwei Zheng³ , Yadong Chang^{1,2},
and Xutao Li^{1,2}

¹ Key Lab of Digital Signal and Image Processing of Guangdong Province,
Shantou University, Shantou 515063, China

{21yjchen1,15zlyang3,21ydchang,lixxt}@stu.edu.cn

² Department of Electronic Engineering, Shantou University, Shantou 515063, China

³ School of Mathematics and Big Data, Foshan University, Foshan 52800, China
alex.w.zheng@hotmail.com

Abstract. Point cloud classification is a fundamental but still challenging task in 3-D computer vision. The main issue is that learning representational features from initial point cloud objects is always difficult for existing models. Inspired by the Transformer, which has achieved successful performance in the field of natural language processing, we propose a purely attention-based network, named PointFormer, for point cloud classification. Specifically, we design a novel simple point multiplicative attention mechanism. Based on that, we then construct both a local attention block and a global attention block to learn fine geometric features and overall representational features of the point cloud, respectively. Consequently, compared to the existing approaches, PointFormer has superior perception of local details and overall contours of the point cloud objects. In addition, we innovatively propose the Graph-Multiscale Perceptual Field (GMPF) testing strategy that can significantly improve the overall performance of the proposed PointFormer. We have conducted extensive experiments on the real-world dataset ScanObjectNN and the synthetic dataset ModelNet40. The results show that the PointFormer has stronger robustness and achieves highly competitive performance compared to other state-of-the-art approaches. The code is available at <https://github.com/Yi-Jun-Chen/PointFormer>.

Keywords: Point cloud classification · Attention mechanism · Feature extraction

1 Introduction

3-D vision is widely used in augmented reality, robotics, autonomous driving and many other fields. Voxels, meshes and point clouds can all be applied to effectively represent 3D data. In contrast to voxels and meshes, point clouds preserve

the original geometric information of 3D objects and are easy to collect. Therefore, it is more suitable for 3D object classification which is a fundamental and still challenging task in 3D vision. However, point clouds are irregular embeddings in continuous space which is different from 2D images. Therefore, existing methods are still challenging for learning representational features in point cloud classification task.

To address this challenge, much work [16, 23] has been inspired by the successful application of convolutional neural networks to 2D computer vision, where 3D objects are projected onto different 2D planes and then the 2D convolution operation is performed. Other methods voxelize the point cloud and then apply 3D discrete convolution in 3D space. Unexpectedly, the performance of these approaches is largely limited by the computational and memory costs. In addition, the transformation of point clouds, such as voxelization and projection, leads to the loss of information, which is detrimental to the processing of point clouds. PointNet [3] is a pioneer in the direct manipulation of point cloud data. It extracts the features of point clouds through shared multilayer perceptron (MLP) and a max-pooling operation, achieving permutation invariance in point cloud processing and impressive classification results. However, PointNet ignores local structure information, which is proven to be important for feature learning in visual recognition. To better capture the local information of point cloud objects, some works [4, 27] introduce attention mechanisms to enhance the local representation of features. Unfortunately, they usually have expensive overhead on memory and arithmetic power. Consequently, it is important to design networks with superior perception of the fine and overall geometric structure of point clouds while maintaining a relatively light weight.

To solve this problem, we propose a purely attention-based network, named PointFormer, for point clouds classification. In detail, we propose a novel and simple point multiplicative attention mechanism. Unlike traditional methods [3, 17, 29] that aggregate information from each point indiscriminately through pooling operations, the point multiplicative attention mechanism treats each point in point cloud differently to extract more discriminative features. Based on this, we then construct local attention block and global attention block that enables the network to have excellent perception of the local fine structure and overall shape of the point cloud. In addition, we propose the Graph-Multiscale Perceptual Field (GMPF) testing strategy for the first time to improve the overall performance of the PointFormer. In summary, our main contributions are displayed below.

- We design a point multiplicative attention mechanism with high expressiveness for point cloud objects. It applies shared multilayer perceptrons (MLP) to learn the encoding of points, which maintains low complexity while satisfying permutation invariance for point cloud processing.
- Based on the point multiplicative attention mechanism, we design both a local attention block and a global attention block for building PointFormer, which enables our model to have superior dual perception of local details and overall contours of point cloud.

- We propose a Graph-Multiscale Perceptual Field (GMPF) testing strategy, which greatly improves the performance of the model. Furthermore, as a general strategy, it can be easily transferred to other networks.
- Extensive experiments on real-world datasets ScanObjectNN and synthetic datasets ModelNet40 show that our proposed has strong robustness and superior classification performance.

2 Related Work

Unlike 2D images which have a regular grid structure, 3D point clouds have spatial continuity and irregularities that preclude direct applications of the existing deep learning models. To overcome this challenge, many approaches have been proposed, which can be broadly classified into three types: multi-view methods, voxel-based methods and point-based methods. We summarise the main features and limitations of these approaches as follows:

- 1) Multi-view Methods: Considering the success of CNNs in 2D images, these methods [6, 9, 23] project 3D point cloud objects onto multiple 2D planes. Then, convolutional neural networks are applied to perform feature extraction. Finally, the aggregation of multi-view features is performed to accurately classify the point cloud. For example, MVCNN [23] is a pioneering network that has achieved impressive results by aggregating multi-scale information through max-pooling operations. However, these methods may result in loss of information during the projection process. At the same time, these approaches usually lead to huge memory and arithmetic overheads because they do not make good use of the sparsity of point clouds.
- 2) Voxel-based Methods: voxel-based approaches regularize irregular point clouds by 3D voxelization [15, 22] and then perform a 3D convolution operation. While these methods can achieve impressive results, the cubic increase in the number of grids can lead to an explosion in computation and memory overhead. To alleviate this problem, OctNet [20] uses shallow octrees to improve computational efficiency, but it is still a computationally intensive approach. In summary, these approaches not only consume computational and memory resources greatly, but also raise information loss during the voxelization process.
- 3) Point-based Methods: Instead of transforming point clouds into other data domains as the previous methods do, the point-based approaches directly use points as input for feature extraction. PointNet [3], as a pioneer model, realizes the real sense of a neural network acting directly on a point cloud. It is implemented by shared MLPs and symmetric operations (e.g. max-pooling) to satisfy the permutation invariance of point cloud processing. However, PointNet operates on each point individually, so it does not have the ability to extract local geometric features. To solve this problem, PointNet++ [17] designs a series of operations for furthest point sampling and grouping, hierarchically aggregating the features of neighbouring nodes. DGCNN [29] makes the local information spread to the global sufficiently by k-nearest neighbor

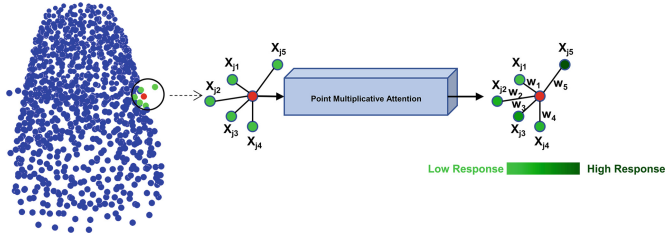


Fig. 1. The red point is selected as the central node, and its k -nearest neighbors ($k=5$ in the figure) are selected to form the k -NN graph. Then, neighboring nodes are weighted by the point multiplicative attention mechanism. The different depths of the colors respond to the magnitude of the weights. (Color figure Online)

(k -NN) composition and dynamic graph update between each layer, which learns the local geometric information of the point cloud very well. However, they uniformly use symmetric pooling operations such as max-pooling to retain the most significant elements, which leads to loss of information and is detrimental to the fine perception of point cloud objects. Other approaches [4, 36] enhance the representation of local information by introducing attention. However, their large number of parameters and computational effort leads to a huge overhead on computational resources.

In contrast, as a point-based approach, our PointFormer has excellent dual perception of the local fine geometry and the overall contour of the point cloud. At the same time, our model is able to maintain a relatively light weight due to the proposed novel and simple point multiplicative attention mechanism. In addition, the proposed Graph-Multiscale Perceptual Field (GMPF) testing strategy gives PointFormer the ability to sense multi-scale information, thus greatly improving the performance of the model.

3 Our Approach

In this section, we first review the general formula for self-attention. Then, the point multiplicative attention mechanism is proposed for constructing our local attention block and global attention block. Finally, the complete framework of PointFormer and the Graph-Multiscale Perceptual Field(GMPF) testing strategy for model performance enhancement is presented.

3.1 Scalar and Vector Self-attention

Self-attention is a set operator [36] which can be classified into two types: scalar attention [27] and vector attention [35]. $\mathcal{X} = \{\mathbf{x}_i\}_i$ denotes the set of feature vectors. The scalar attention can be formulated as follows:

$$\mathbf{x}'_i = \sum_{\mathbf{x}_j \in \mathcal{X}} S\left(M_1(\mathbf{x}_i)^\top M_2(\mathbf{x}_j)\right) M_3(\mathbf{x}_j), \quad (1)$$

where \mathbf{x}'_i is the updated feature. M_1 , M_2 and M_3 correspond to different mappings such as projections or MLPs. S is a normalization operation such as softmax. Scalar attention is actually a projection of feature vectors into different feature spaces and then the corresponding attention coefficients are found by calculating the inner product.

In vector attention, the formulation is slightly different:

$$\mathbf{x}'_i = \sum_{\mathbf{x}_j \in \mathcal{X}} S(M_4(M_1(\mathbf{x}_i), M_2(\mathbf{x}_j))) \odot M_3(\mathbf{x}_j), \quad (2)$$

M_4 represents a mapping (e.g. MLP) like M_1 , M_2 , M_3 . Vector attention replaces the operation of inner product in scalar attention by introducing more learnable parameters in M_4 . Both scalar and vector attention are set operator. Therefore, it satisfies the permutation invariance of point cloud processing and is well suited for processing point clouds.

3.2 Point Multiplicative Attention Mechanism

From the viewpoint of formulation and computation, scalar attention is simpler, but often less effective than vector attention. Conversely, vector attention is more effective but more complex. To reach a reliable balance between these mechanisms, we propose our point multiplicative attention mechanism:

$$\mathbf{x}'_i = \sum_{\mathbf{x}_j \in \mathcal{X}} S(L_3^1(L_1(\mathbf{x}_i) + L_2(\mathbf{x}_j))) L_2(\mathbf{x}_j), \quad (3)$$

where L_1, L_2 denote different single fully connected layers respectively. L_3^1 denotes a single fully connected layer with an output dimension of 1. As we can see, to reduce the complexity of vector attention, we replace all the MLPs in Eq. 3 with a linear layer. In particular, we replace the different MLPs M_2 , M_3 in Eqs. 1 and 2 with the same single fully connected layer L_2 . At the same time, unlike GAN [28], we use the vector addition instead of vector concatenation in Eq. 3 to reduce the dimension of features. Moreover, subject to scalar attention, we only output an attention coefficient for each neighbor feature \mathbf{x}_j through the fully connected layer L_3^1 and then perform a weighted summation. This is the reason why our attention is called point multiplicative attention.

3.3 Local Attention Block

As point multiplicative attention follows a similar formulation as scalar and vector attention, it is still a set operator. We construct the k-nearest neighbor (k-NN) graph [29] for each node of the point cloud, as shown in Fig. 1. The features of the central node and the neighbor nodes constitute the set of feature vectors for the attention operation. We construct our local attention block by acting the point multiplicative attention mechanism on the k-NN graph of each point (as presented in Fig. 1).

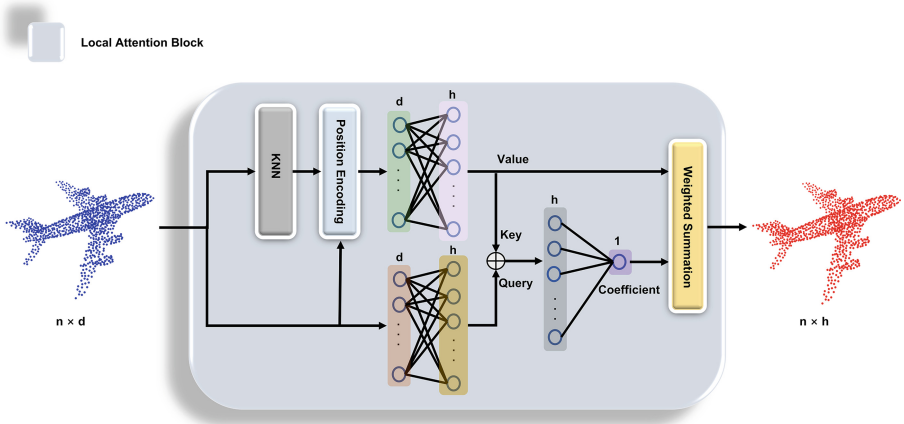


Fig. 2. Framework diagram of the local attention block. Input: Initial point cloud object features (blue). Output: Updated point cloud object features. The local attention block can be viewed as a d -dimensional to h -dimensional feature mapping. (Color figure online)

Our local attention block providing a better perception of local details of the point cloud objects. For example, in Fig. 1, the neighbor \mathbf{x}_{j5} represents the sampled points from the rearview mirror of the car, while the other neighbors are the sampled points from the body. Point multiplicative attention mechanism can place on neighbor \mathbf{x}_{j5} with a larger attention coefficient, thus paying more attention to this particular neighbor points while treating other neighbors differently.

Position Encoding: From the viewpoint of graph neural networks [7, 34], attention is an information aggregation operator. Specifically, we learn the aggregation feature of neighboring points \mathbf{x}_j to centroid \mathbf{x}_i by L_2 in Eq. 3. Then, the aggregation of information is completed by weighted summation of the corresponding attention coefficients. Therefore, the linear layer L_2 can be viewed as a fit of the abstract function $f(\cdot)$ between \mathbf{x}_i and \mathbf{x}_j . However, considering only the features \mathbf{x}_j of neighbour nodes in L_2 may not make good use of the initial geometric information of the point cloud. In order to take the higher order features into considerations, we use \mathbf{x}_i , \mathbf{x}_j , $(\mathbf{x}_i - \mathbf{x}_j)$ and $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$ as inputs to L_2 . This means that we use higher order difference terms $(\mathbf{x}_i - \mathbf{x}_j)$ and $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$ to retain high frequency detail information.

Finally, our local attention block can be formulated as:

$$\mathbf{x}'_i = \sum_{\mathbf{x}_j \in \mathcal{X}} S(L_3^1(L_1(\mathbf{x}_i) + L_2(PE(\mathbf{x}_j)))) L_2(PE(\mathbf{x}_j)), \quad (4)$$

where PE denotes our position encoding strategy. The framework of the local attention block is illustrated in Fig. 2.

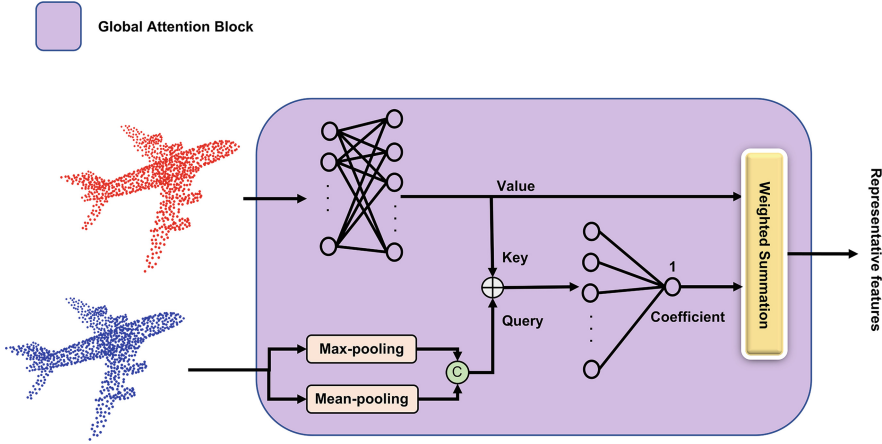


Fig. 3. Architecture of global attention block. The input is a point cloud carrying high-dimensional features $[n \times h]$, and the output is a representation vector of the point cloud $[n \times 1]$.

3.4 Global Attention Block

Local attention block provides excellent perception of local details, but lacks overall representation of the point cloud. To address this problem, global attention block is designed to extract highly representational features from point cloud objects. In detail, our global attention block is designed to apply point multiplicative attention to the entire point cloud object that carrying high-dimensional features, i.e., the red point cloud output from the local attention block in Fig. 2. Specifically, we concatenate the max-pooling and mean-pooling of the point cloud 3D coordinates to obtain the query vector. If $\mathcal{X} = \{\mathbf{x}_j\}_j$ denotes the set of features of all points of a point cloud object, the global attention block can be formulated as:

$$\mathbf{y} = \sum_{\mathbf{x}_j \in \mathcal{X}} S(L_3^1(L_1(query) + L_2(\mathbf{x}_j))) L_2(\mathbf{x}_j), \tag{5}$$

where \mathbf{y} denotes the representation vector of the point cloud. The complete structure of global attention block is presented in Fig. 3. The global attention block can be assigned an attention level for each point. This mechanism of treating each point differently allows our model to have superior perception of the overall contour of the point cloud. We will further visualize it in the experimental section.

3.5 Framework of PointFormer

In summary, we use four local attention blocks and one global attention block to build the PointFormer, as shown in Fig. 4. The output dimensions are set

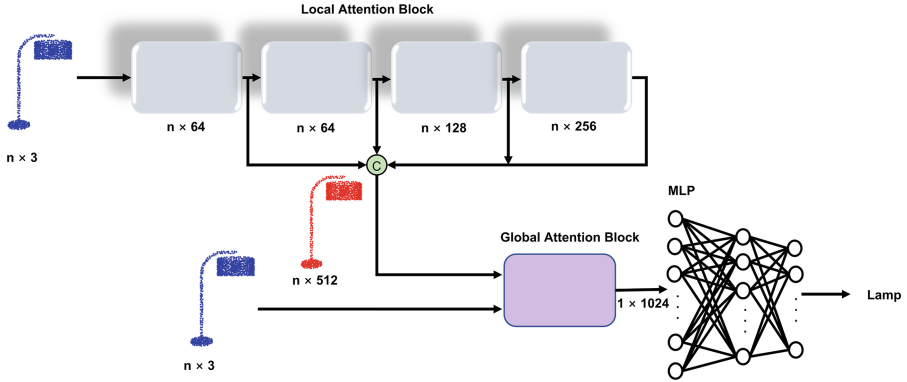


Fig. 4. Architecture of PointFormer.

to (64, 64, 128, 256) and (1024), respectively. Meanwhile, shortcut connections are applied to extract multi-scale features. Finally, the classification results are obtained by Multi-layer Perceptron (MLP).

Hierarchical Position Encoding (HPE): To maximize the initial geometric information of the point cloud while maintaining the lightweight of the model, our position encoding strategy in local attention block is further designed to use \mathbf{x}_i , \mathbf{x}_j , $(\mathbf{x}_i - \mathbf{x}_j)$ and $\|\mathbf{x}_i - \mathbf{x}_j\|_2^2$ as the input of L_2 only in the first local attention block. However, in the subsequent three blocks, we just use \mathbf{x}_j and first-order difference term $(\mathbf{x}_i - \mathbf{x}_j)$ [29]. We refer to this location encoding strategy as Hierarchical Position Encoding (HPE).

3.6 Graph-Multiscale Perceptual Field (GMPF) Testing Strategy

In order to improve the performance while without introducing any burden (e.g., parameters, inference time), we design a novel Graph-Multiscale Perceptual Field (GMPF) testing strategy. Briefly, in the training phase, we construct the k-nearest neighbor (k-NN) graph using a specific number of neighbors (e.g., $k = 5$ in Fig. 1), while in the testing phase, multi-scale k-NN graphs are used for testing (e.g., $k=3, 5, 7$). Equivalently, the GMPF strategy is applied to learn a model on a fixed-scale k-NN graph while observing the information on the multi-scale graph during testing as shown in Fig. 5. In the experimental section, we will further show that this strategy can greatly improve the robustness and overall performance of the model. Moreover, as a general strategy, it can be naturally transferred to other graph-based networks.

4 Experiments

In this section, we first provide the experimental setup for evaluation. We then comprehensively evaluate PointFormer on ScanObjectNN [26] and ModelNet40

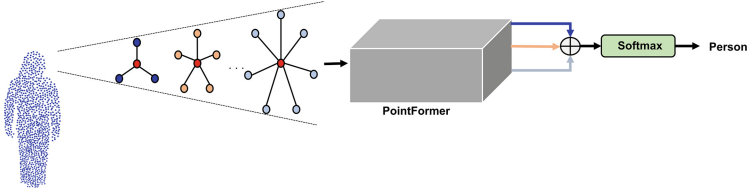


Fig. 5. The Graph-Multiscale Perceptual Field testing strategy. It can also be analogous to a person identifying objects from different ranges of vision from far to near. In reality, this can greatly improve human recognition when objects are obscured or dimly lit.

Table 1. Classification Results (%) on ScanObjectNN dataset. The Accuracy gap (last column) represents the difference between Mean Class Accuracy and Overall Accuracy (%).

Method	Mean Class Accuracy	Overall Accuracy	Accuracy gap
3DmFV [2]	58.1	63.0	4.9
PointNet [3]	63.4	68.2	4.8
SpiderCNN [32]	69.8	73.7	3.9
PointNet++ [17]	75.4	77.9	2.5
DGCNN [29]	73.6	78.1	4.5
PointCNN [11]	75.1	78.5	3.4
BGA-DGCNN [26]	75.7	79.7	4.0
BGA-PN++ [26]	77.5	80.2	2.7
DRNet [18]	78.0	80.3	2.3
GBNet [19]	77.8	80.5	2.7
PointFormer	78.9	81.1	2.2

[30] benchmarks. Finally, we conducted a large number of careful and reasonable ablation experiments to prove the rationality and validity of each component of the proposed network.

4.1 Experimental Setup

- 1) Training: We implement PointFormer using PyTorch and employ SGD optimizer with an initial learning rate of 0.1 and the momentum of 0.9 for training. We use the cosine annealing [14] to reduce the learning rate until 0.001. The batch size is 36, number of neighbors is $k=20$ and we do not use batch normalization decay. The epochs on the ScanObjectNN dataset and the ModelNet40 dataset are 350 and 250, respectively.
- 2) Testing: In our testing, we use the GMPF testing strategy. Specifically, we set the multi-scale k -NN graph to $k=(15,16,20,25,28)$ on the ScanObjectNN dataset and $k=(15,20,28)$ on the ModelNet40 dataset respectively. The selection will be further explained in the experimental section.

Table 2. Classification Results (%) on ModelNet40 dataset. (coords: 3-dimensional coordinates, norm: point normal vector, voting: multivotes test method, k: $\times 1024$, -: unknown)

Method	Input	#point	Mean Class Accuracy	Overall Accuracy
ECC [21]	coords	1k	83.2	87.4
PointNet [3]	coords	1 k	86.0	89.2
SCN [31]	coords	1 k	87.6	90.0
Kd-Net [8]	coords	1 k	–	90.6
PointCNN [11]	coords	1 k	88.1	92.2
PCNN [1]	coords	1 k	–	92.3
DensePoint [12]	coords	1 k	–	92.8
RS-CNN [13]	coords	1 k	–	92.9
DGCNN [29]	coords	1 k	90.2	92.9
DGCNN [29]	coords	2 k	90.7	93.5
KP-Conv [25]	coords	1 k	–	92.9
PointASNL [33]	coords	1 k	–	92.9
SO-Net [10]	coords	2 k	87.3	90.9
SO-Net [10]	coords + norm	5 k	90.8	93.4
RGCNN [24]	coords + norm	1 k	87.3	90.5
PointNet++ [17]	coords + norm	5 k	-	91.9
SpiderCNN [32]	coords + norm	5 k	–	92.4
DensePoint [12]	coords + voting	1 k	–	93.2
RS-CNN [13]	coords + voting	1 k	–	93.6
PCT [5]	coords	1 k	–	93.2
Point Transformer [36]	coords	1 k	90.6	93.7
GB-Net [19]	coords	1 k	91.0	93.8
PointFormer	coords	1 k	90.7	93.7

4.2 Classification Results

- 1) Classification on ScanObjectNN: ScanObjectNN consists of approximately 15,000 objects, which belong to 15 categories. Table 1 shows the classification results comparison between PointFormer and each competitive benchmark on ScanObjectNN. It can be seen that our model achieves 78.9% mean class accuracy and 81.1% overall class accuracy, which is 0.9% and 0.6% higher than the previous best results, respectively. Note that the number of parameters of PointFormer is 3.99 M and FLOPs is 3.48 G, while for GBNet [19] is of 8.78 M and 11.57 G, respectively, which is much larger than our model. However, we achieve a superior performance than it, which means we have earned a better balance between model complexity and performance. Besides, our PointFormer attains the smallest gap between mean class accuracy and overall accuracy. This displays that our network has superior performance for each categorie, exhibiting better inter-class robustness.

The objects in ScanObjectNN contain many distortion points, background points and missing regions [26], which poses a serious challenge to the robustness of the model. However, the local attention block and global attention block in PointFormer bring an effective solution to this problem. For example, in the local attention block, the attention weights the features of each neighbor node, which allows the model to learn to discard more useless nodes during training. Moreover, when the features are finally aggregated, our global attention block can even abandon unwanted nodes directly by setting the coefficient to low response. *More importantly, our GMPF testing strategy can fully extract useful information from the multi-scale k-NN graph.* This can greatly improve the noise immunity of the model when the dataset is not pure. This means that our model is able to eliminate the detrimental effects of more noise points. So PointFormer is able to demonstrate booming robustness in unclean real-world datasets.

Table 3. Graph-Multiscale Perceptual Field testing strategy (GMPF) effectiveness analysis on the ScanObjectNN and ModelNet40 datasets. We are comparing mean class and overall class accuracy (%)

	ScanObjectNN		ModelNet40	
	Mean Class Acc.	Overall Acc.	Mean Class Acc.	Overall Acc.
PointFormer	77.2	80.1	90.5	93.6
PointFormer (GMPF)	78.9	81.1	90.7	93.7

2) Classification on ModelNet40: This dataset, specifically, contains 12,311 synthetic pure CAD models in which the point cloud data is divided into 40 categories. Typically, 9843 models are used for training and 2468 models are reserved for testing in our experiment. Table 2 presents the highly competitive classification result for our network (mean class accuracy: 90.7% and overall accuracy: 93.7%). Note that our model uses only the 1k 3D coordinates of the point cloud as input, but carries out even better results than some approaches that use more information. As can be observed in Table 2, DGCNN [29] takes 2k points as input and achieves 90.7% average class accuracy and 93.5% overall accuracy. RGCNN [24] additionally used the normal vector of points as input to attain 90.5% overall accuracy. SO-Net [10] even used 5k points and normal vectors, achieving an average classification accuracy of 90.8% and an overall accuracy of 93.4%. Moreover, our model also outperforms the models that employ the voting strategy such as RSCNN [13] and DensePoint [12].

It is worth noting that although the classification performance of PointFormer on ModelNet40 is slightly worse than GB-Net [19], our model is far lighter and more robust than GB-Net as analyzed earlier, which implies that our PointFormer is more practical in real-world applications.

4.3 PointFormer Design Analysis

To demonstrate the soundness and validity of our model design, we conduct comparative and ablation experiments in this section, as well as conducting the visualization and analysing the complexity of the model.

- 1) Effectiveness of GMPF testing strategy: GMPF is our innovative work, which does not bring any parametric quantity to the model, but greatly improves the robustness and performance of our model. We verify the effectiveness of this strategy by comparing the performance of the model with GMPF and the model without GMPF on the ScanObjectNN and ModelNet40 dataset. The results are shown in Table 3. As can be seen, on the real-world ScanObjectNN dataset, the GMPF strategy *improved the average accuracy and overall accuracy by 1.7% and 1%, respectively*, which is a stunning result with significant relevance! Besides, the GMPF strategy on clean synthetic ModelNet40 dataset also earns good gains. This strategy is like learning at a specific angle while recognizing point cloud objects at multiscale related angles. The effect may not be as dramatic when the data is very pure, but when faced with realistic unclean datasets, GMPF can greatly improve the robustness and overall performance of the network.

Table 4. The effects of different choices of multiscale k-NN graph on the ScanObjectNN datasets. We are comparing mean class accuracy and overall accuracy (%).

	k = (20)	k = (20, 28)	k = (15, 20, 25)	k = (15, 20, 28)	k = (15, 16, 20, 25, 28)
Mean Class Acc	77.2	77.9	78.8	78.7	78.9
Overall Acc	80.1	80.2	80.9	80.9	81.1

Table 5. The results (%) of different HPE strategies. We are comparing the overall class accuracy on the ModelNet40 dataset. L_2 : the l_2 paradigm (3D Euclidean distance) of x_i-x_j , x_i : center vertex, x_j : neighborhood nodes, Feature Dimension: the input channel of the four local attention block respectively.

Model	Hierarchical Position Encoding	Feature Dimension	Overall Accuracy
A	$4^*(x_i, x_i-x_j)$	(6, 128, 128, 256)	93.3
B	(x_i, x_j, L_2^2) and $3^*(x_i, x_j)$	(7, 128, 128, 256)	93.1
C	(x_i, x_j, x_i-x_j, L_2) and $3^*(x_i, x_j, x_i-x_j)$	(10, 192, 192, 384)	92.9
D	$(x_i, x_j, x_i-x_j, L_2^2)$ and $3^*(x_i, x_i-x_j)$	(10, 128, 128, 256)	93.7

On the ScanObjectNN and ModelNet40 datasets, the multiscale k-nearest neighbor graphs were selected as $k=(15,16,20,25,28)$ and $k=(15,20,28)$, respectively. In Table 4, we compare the effects of different choices of k-NN graph on the performance of the PointFormer on the ScanObjectNN dataset. As we can see, our selection based on which makes the testing result best. Similarly, on the ModelNet40 dataset, we do the same.

- 2) Effectiveness of Hierarchical Position Encoding (HPE): We verified the effectiveness of HPE and compared different encoding strategies on ModelNet40 dataset. The results are presented in Table 5. Compared to models A and B, our model D retains and utilises more point cloud geometry information. At the same time, it does not increase the size of the model and introduce too many parameters. The model C, on the other hand, exponentially increases the FLOPs and the size of the model when the features are high-dimensional. This greatly boost the risk of overfitting. Thus, it can be seen that our Hierarchical Position Relation strikes a good balance between model complexity and maximizing the use of the original geometric information in point cloud object.
- 3) Network Complexity: We analyze the network complexity of PointFormer and other advanced approaches by comparing the number of parameters and FLOPs on ModelNet40 dataset. For the sake of fairness, we standardize the testing conditions (GeForce RTX 3090, batch size = 1). The results are displayed in Table 6. As we can see, the complexity of our model is slightly higher than that of the traditional DGCNN approach, but compared with current Attention-based methods such as (2) and (4), our model is much lighter and achieves better results. It is worth noting that, for simplicity, we perform the comparison on a clean synthetic dataset ModelNet40, but the greater strength of our network is the strong robustness demonstrated on the impure real-world dataset ScanObjectNN. However, the more complex approaches (2) and (4) do not necessarily stand up to the test in this case. At the same time, compared to GBNet [19], PointFormer is much lighter and has better robustness, as mentioned in the previous analysis.

Table 6. Complexity analysis of classification network on the ModelNet40 dataset.

Method	Param	FLOPs	Overall acc. (%)
(1)DGCNN [29]	1.81 M	2.72 G	92.9
(2)GBNet [19]	8.78 M	11.57 G	93.8
(3)Point Transformer [36]	9.58 M	18.41 G	93.7
(4)Point Transformer [4]	21.67 M	4.79 G	92.8
PointFormer	3.99 M	3.48 G	93.7

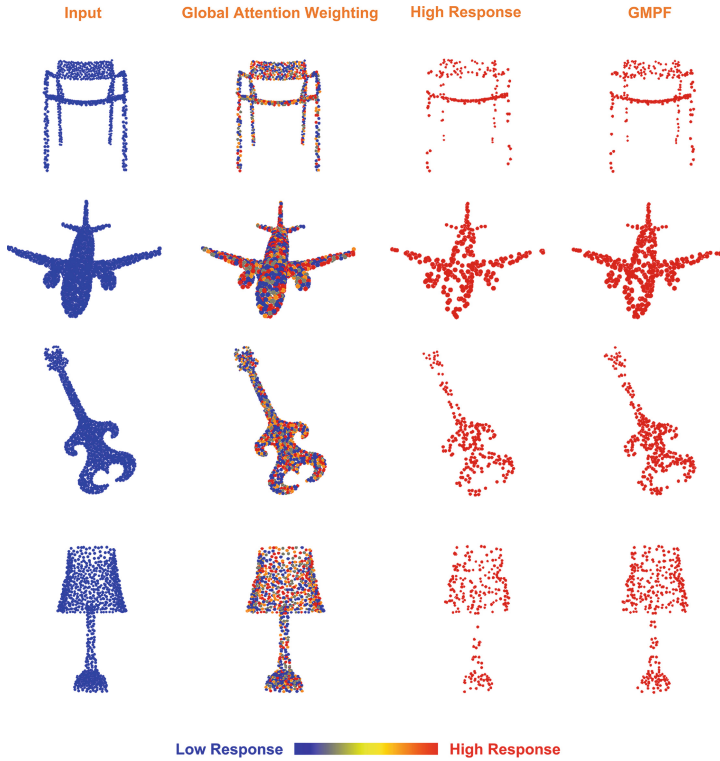


Fig. 6. Visualisation of the importance of each point in the point cloud. Each point is coloured by the attention factor. Blue represents the lowest attention and red the highest. (Color figure online)

- 4) Visualization: To illustrate PointFormer’s excellent perception of point cloud objects, we visualize the attention factor of the global attention block at each point. As shown in Fig. 6, PointFormer focuses more on recognizable contour points in the point cloud, while treats other unimportant points differently. In the case of airplane, the points that PointFormer pays the most attention to are distributed in various parts of the airplane, such as wings, tail, fuselage, etc., which well form a recognizable outline of an airplane. However, PointFormer can selectively ignore some optional points, as shown in the blue points in Fig. 6. In addition, our testing strategy (GMPF) allows the model to be more perceptive of the overall contours of the point cloud. As presented in the last column of Fig. 6, the highly responsive points provide a more complete and accurate representation of the point cloud shape.

5 Conclusion

In this paper, we present a purely attention-based network, PointFormer, for the point cloud classification task. Specifically, we design the local attention block, which enables the network to have the ability in fine local perception with finer-grained local geometric features of the point cloud. At the same time, we designed the global attention block, so that PointFormer can perceive point cloud objects as a whole to improve the performance in classification. In addition, the general GMPF strategy greatly improves the performance and robustness of the model while easily transferred to other models. The highly competitive results on the synthetic dataset ModelNet40 and the real-world dataset ScanObjectNN demonstrate the outstanding advantages and a promising performance of the proposed model. We believe that PointFormer is a superior feature extractor, not only for classification tasks, but also for part segmentation, scene segmentation and even point cloud completion.

Acknowledgement. This work was supported by the National Natural Science Foundation of China (No. 61471229 and No. 61901116), the Natural Science Foundation of Guangdong Province (No. 2019A1515011950), the Guangdong Basic and Applied Basic Research Foundation (No. 2019A1515010789 and No. 2021A1515012289), and in part by the Key Field Projects of Colleges and Universities of Guangdong Province (No. 2020ZDZX3065), and in part by Shantou University Scientific Research Foundation for Talents under Grant NTF19031.

References

1. Atzmon, M., Maron, H., Lipman, Y.: Point convolutional neural networks by extension operators. CoRR abs/1803.10091 (2018). <http://arxiv.org/abs/1803.10091>
2. Ben-Shabat, Y., Lindenbaum, M., Fischer, A.: 3DmFV: three-dimensional point cloud classification in real-time using convolutional neural networks. IEEE Robotics and Automation Letters **3**(4), 3145–3152 (2018). <https://doi.org/10.1109/LRA.2018.2850061>
3. Charles, R.Q., Su, H., Kaichun, M., Guibas, L.J.: PointNet: deep learning on point sets for 3D classification and segmentation. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 77–85 (2017). <https://doi.org/10.1109/CVPR.2017.16>
4. Engel, N., Belagiannis, V., Dietmayer, K.: Point transformer. IEEE Access **9**, 134826–134840 (2021). <https://doi.org/10.1109/ACCESS.2021.3116304>
5. Guo, M., Cai, J., Liu, Z., Mu, T., Martin, R.R., Hu, S.: PCT: point cloud transformer. Comput. Vis. Media **7**(2), 187–199 (2021)
6. Kanezaki, A., Matsushita, Y., Nishida, Y.: RotationNet: joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition(CVPR), pp. 5010–5019 (2018). <https://doi.org/10.1109/CVPR.2018.00526>

7. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings (2017). <https://openreview.net/forum?id=SJU4ayYgl>
8. Klokov, R., Lempitsky, V.: Escape from cells: Deep KD-networks for the recognition of 3D point cloud models. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 863–872 (2017). <https://doi.org/10.1109/ICCV.2017.99>
9. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: PointPillars: fast encoders for object detection from point clouds. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12689–12697 (2019). <https://doi.org/10.1109/CVPR.2019.01298>
10. Li, J., Chen, B.M., Lee, G.H.: SO-Net: self-organizing network for point cloud analysis. In: 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9397–9406 (2018). <https://doi.org/10.1109/CVPR.2018.00979>
11. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: convolution on X-transformed points. In: Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3–8 December 2018, Montréal, Canada, pp. 828–838 (2018)
12. Liu, Y., Fan, B., Meng, G., Lu, J., Xiang, S., Pan, C.: DensePoint: learning densely contextual representation for efficient point cloud processing. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 5238–5247 (2019). <https://doi.org/10.1109/ICCV.2019.00534>
13. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8887–8896 (2019). <https://doi.org/10.1109/CVPR.2019.00910>
14. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with warm restarts. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, 24–26 April 2017, Conference Track Proceedings (2017)
15. Maturana, 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), pp. 922–928 (2015). <https://doi.org/10.1109/IROS.2015.7353481>
16. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view cnns for object classification on 3D data. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5648–5656 (2016). <https://doi.org/10.1109/CVPR.2016.609>
17. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
18. Qiu, S., Anwar, S., Barnes, N.: Dense-resolution network for point cloud classification and segmentation. In: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 3812–3821 (2021). <https://doi.org/10.1109/WACV48630.2021.00386>
19. Qiu, S., Anwar, S., Barnes, N.: Geometric back-projection network for point cloud classification. *IEEE Trans. Multimedia* **24**, 1943–1955 (2022). <https://doi.org/10.1109/TMM.2021.3074240>
20. Riegler, G., Ulusoy, A.O., Geiger, A.: OctNet: learning deep 3D representations at high resolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6620–6629 (2017). <https://doi.org/10.1109/CVPR.2017.701>

21. Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 29–38 (2017). <https://doi.org/10.1109/CVPR.2017.11>
22. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 190–198 (2017). <https://doi.org/10.1109/CVPR.2017.28>
23. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.: Multi-view convolutional neural networks for 3d shape recognition. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 945–953 (2015). <https://doi.org/10.1109/ICCV.2015.114>
24. Te, G., Hu, W., Zheng, A., Guo, Z.: RGCNN: regularized graph CNN for point cloud segmentation. In: Boll, S., et al. (eds.) 2018 ACM Multimedia Conference on Multimedia Conference, MM 2018, Seoul, Republic of Korea, 22–26 October 2018, pp. 746–754. ACM (2018). <https://doi.org/10.1145/3240508.3240621>
25. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.: KPConv: flexible and deformable convolution for point clouds. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 6410–6419 (2019). <https://doi.org/10.1109/ICCV.2019.00651>
26. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, T., Yeung, S.K.: Revisiting point cloud classification: a new benchmark dataset and classification model on real-world data. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1588–1597 (2019). <https://doi.org/10.1109/ICCV.2019.00167>
27. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4–9 December 2017, Long Beach, CA, USA, pp. 5998–6008 (2017)
28. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. CoRR abs/1710.10903 (2017)
29. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. *ACM Trans. Graph.* **38**(5), 3326362 (2019). <https://doi.org/10.1145/3326362>
30. Wu, Z., et al.: 3d shapeNets: a deep representation for volumetric shapes. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1912–1920 (2015). <https://doi.org/10.1109/CVPR.2015.7298801>
31. Xie, S., Liu, S., Chen, Z., Tu, Z.: Attentional shapeContextNet for point cloud recognition. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4606–4615 (2018). <https://doi.org/10.1109/CVPR.2018.00484>
32. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Yu.: SpiderCNN: deep learning on point sets with parameterized convolutional filters. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11212, pp. 90–105. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01237-3_6
33. Yan, X., Zheng, C., Li, Z., Wang, S., Cui, S.: PointASNL: robust point clouds processing using nonlocal neural networks with adaptive sampling. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5588–5597 (2020). <https://doi.org/10.1109/CVPR42600.2020.00563>
34. Yu, Z., Zheng, X., Yang, Z., Lu, B., Li, X., Fu, M.: Interaction-temporal GCN: a hybrid deep framework for covid-19 pandemic analysis. *IEEE Open J. Eng. Med. Biol.* **2**, 97–103 (2021). <https://doi.org/10.1109/ojemb.2021.3063890>

35. Zhao, H., Jia, J., Koltun, V.: Exploring self-attention for image recognition. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10073–10082 (2020). <https://doi.org/10.1109/CVPR42600.2020.01009>
36. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 16259–16268 (2021)