MPCT: Multiscale Point Cloud Transformer With a Residual Network

Yue Wu, *Member, IEEE*, Jiaming Liu, Maoguo Gong, *Senior Member, IEEE*, Zhixiao Liu, Qiguang Miao, *Senior Member, IEEE*, and Wenping Ma, *Senior Member, IEEE*

***Abstract*—The self-attention (SA) network revisits the essence of data and has achieved remarkable results in text processing and image analysis. SA is conceptualized as a set operator that is insensitive to the order and number of data, making it suitable for point sets embedded in 3D space. However, working with point clouds still poses challenges. To tackle the issue of exponential growth in complexity and singularity induced by the original SA network without position encoding, we modify the attention mechanism by incorporating position encoding to make it linear, thus reducing its computational cost and memory usage and making it more feasible for point clouds. This article presents a new framework called multiscale point cloud transformer (MPCT), which improves upon prior methods in cross-domain applications. The utilization of multiple embeddings enables the complete capture of the remote and local contextual connections within point clouds, as determined by our proposed attention mechanism. Additionally, we use a residual network to facilitate the fusion of multiscale features, allowing MPCT to better comprehend the representations of point clouds at each stage of attention. Experiments conducted on several datasets demonstrate that MPCT outperforms the existing methods, such as achieving accuracies of 94.2% and 84.9% in classification tasks implemented on ModelNet40 and ScanObjectNN, respectively.**

***Index Terms*—Geometric and semantic features, multiscale generation, point cloud transformer, residual network.**

I. Introduction

With the rapid development of 3D sensing technology, 3D point cloud data are appearing in many application areas such as autonomous driving, virtual and augmented reality, and robotics. Driven by deep neural networks, recent 3D works [1], [2], [3], [4], [5], [6], [7], [8] have focused on processing point clouds with learning-based methods. However, unlike images arranged on a regular pixel or grid, point clouds are sets of points embedded in three-dimensional space. This makes 3D point clouds structurally different from images and representationally different from complex 3D data (e.g., grid and voxel data), which have the simplest format but cannot be directly applied to design deep networks for standard tasks in computer vision.

To address this challenge, many deep learning-based methods have become powerful tools for representing 3D point clouds. Guo et al. [9] classified learning-based point cloud methods into multiview methods [10], [11], voxel methods [12], [13], and point methods [14], [15]. Since Qi et al. introduced the multilayer perceptron (MLP) operation in PointNet, point-based methods have become popular, and subsequent methods [16], [17] have used graph context and kernel points to conduct further learning on the basis of MLPs. Although these methods aim to improve point cloud understanding, some issues remain to be considered. **1)** In addition to 3D coordinates, *can we provide more geometric information for feature learning?* **2)** *How can the network learn better representations in the abstract high-level space and while keeping its costs low and efficiency high?* **3)** *How can the network focus on features and connect different features to optimize the output?*

Compared to point- and convolution-based learning methods [17], [18], [19], [20], [21], [22], [23], [24], transformers [25] have recently been applied to text language and image vision tasks, and their performance is impressive in terms of numerous aspects [26], [27]. Prior to this, more research [28], [29], [30], [31], [32], [33] began to study the use of transformers in point clouds. Transformer is a classic decoder-encoder structure that contains three main modules: input (word) embedding, position (sequential) encoding, and self-attention (SA) modules. Interestingly, the series of transformer operations are especially suitable for point cloud processing, as demonstrated by the fact that they can uniquely establish remote dependencies between points. In addition, the core SA module of the network is essentially a set operator, which is independent of the order and distribution of the given data.

However, when applying a transformer to point clouds, the input sequence generated from points of the same size only retains a simple scale feature in the same layer with an MLP.

Manuscript received 25 October 2022; revised 17 February 2023 and 3 August 2023; accepted 1 September 2023. Date of publication 12 September 2023; date of current version 14 February 2024. This work was supported in part by the National Natural Science Foundation of China under Grants 62276200 and 62036006, in part by the Natural Science Basic Research Plan in Shaanxi Province of China under Grant 2022JM-327, and in part by the CAAI-Huawei MINDSPORE Academic Open Fund. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. De-Nian Yang. *(Corresponding author: Maoguo Gong.)*

Yue Wu, Jiaming Liu, and Qiguang Miao are with the School of Computer Science and Technology, Key Laboratory of Collaborative Intelligence Systems, Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: ywu@xidian.edu.cn; ljm@stu.xidian.edu.cn; qgmiao@mail.xidian.edu.cn).

Maoguo Gong is with the School of Electronic Engineering, Key Laboratory of Collaborative Intelligence Systems, Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: gong@ieee.org).

Zhixiao Liu is with the Yantai Research Institute, Harbin Engineering University, Yantai 264006, China (e-mail: robinliu@hrbeu.edu.cn).

Wenping Ma is with the School of Artificial Intelligence, Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China (e-mail: wpma@mail.xidian.edu.cn).

https://github.com/ywu0912/TeamCode.git.

Digital Object Identifier 10.1109/TMM.2023.3312855

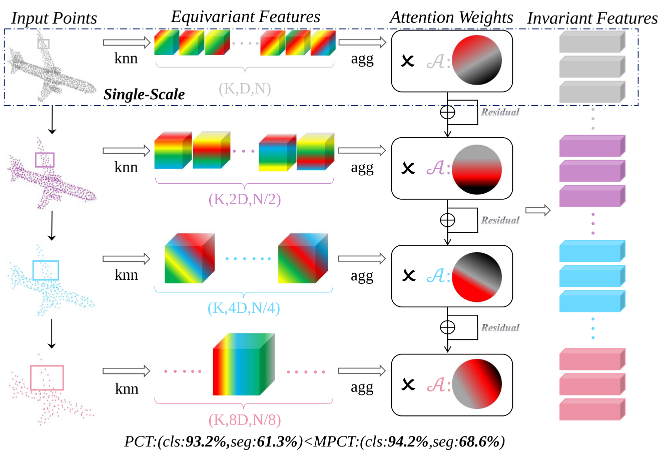


Fig. 1. Multiscale feature fusion module. We present a comparison between PCT with a single-scale and our proposed MPCT with multiscales at the bottom. The top-down transformation has a sampling rate of 1/2.

Point Transformer (PT) [28] was proposed to build local attention for neighborhood and Point Cloud Transformer (PCT) [30] was proposed to use a neighborhood embedding strategy to improve point embeddings. Despite the progress made with PT and PCT, their efficiency is still limited by the original SA module, which requires generating a large attention map and calculating all points, leading to an $\mathcal{O}(N^2D)$ complexity.

On the basis of the previous works, we propose a novel architecture called **M**ulti-scale **P**oint **C**loud **T**ransformer (MPCT), whose key idea is to use the permutation invariance of self-attention to avoid the problems caused by the irregularity of point clouds. As shown in Fig. 1, we are the first to use the proposed novel self-attention mechanism to analyze multistage local neighborhoods and generate multiscale features. In particular, following the k-nearest neighbors (KNN) algorithm, we find the neighbors of a point p_i, which are denoted as $\forall p_{i_k} \in \mathcal{N}(p_i)$. The equivariant feature f_i corresponding to point p_i has a local feature map in that D-dimensional space, which is defined as $\tilde{f}_i = \theta(f_i, \forall f_{i_k}) \in \mathbb{R}^{2D}$, where θ denotes a point feature and its local feature. The fused features are obtained by a local aggregation operation. Then, by feeding the transformed features to the self-attention module, the potential connections between the current features can be established, and new invariant features can then be extracted (Q2). During each transformation, the use of residual networks enables the creation of memory-efficient deep networks. Note that we further enhance the processing of position information (Q1) for each point in the point cloud before each data input step and use the residual structure [34] to maintain network memory (Q3) after each feature generation step.

In summary, the proposed MPCT is straightforward and efficient, relying solely on self-attention and pointwise operations. Reviewing the above three questions, the corresponding contributions of this article are summarized as follows.

- We propose a point enhancement method to upgrade points to high-level geometric information to facilitate self-attention and distinguished-attention.
- We design an expressive point cloud transformer layer that effectively integrates geometric and semantic features and has a low linear complexity of $\mathcal{O}(ND)$.
- We verify that even with a simple residual network connection strategy, effective memory-efficient networks can be built and aid in feature propagation.
- We experiment on multiple datasets in multiple domains, and MPCT achieves results comparable to state-of-the-art methods on multiple competitive benchmarks.

The rest of the article is organized as follows. In Section II, we briefly review the work related to transformers. In Section III, we introduce the preliminary techniques and proposed MPCT architecture. Section IV evaluates our method on five datasets in comparison with several other classic methods and presents the ablation studies. Section V concludes.

II. Related Work

A. Geometric Information and Local Features

Although point cloud data collection is becoming increasingly convenient, compared with regular grids or voxels, these data mainly lack high-level geometric information. In addition to 3D point coordinates, conventional methods [35], [36] estimate more geometric information about a point cloud, such as plane, normal, curvature, etc., but this is limited to the use of CNN-based methods. When [37], [38] designed their the network structures, they tried to combine the relative positions and distances between the points in the encoder. We hope that the learning geometric point features can play a role in the shift from the low-level space to the high-level space.

PointNet++ [15] aggregates local information on the basis of PointNet [14] with farthest point sampling and ball query grouping. It has been shown that local features are effective. MS-GraphSIM [39] uses a multiscale local feature fusion network to assess the quality of point clouds, and PM-BVQA [40] evaluates colored point clouds using a joint color-geometry module. Merigot et al. [36] used the KNN algorithm to be more efficient, but such methods are strongly affected by local neighbors, especially when artificial noise is added to the training data. To overcome these problems, we apply point enhancement and multiscale local neighbors to reduce the possible deviations during feature learning.

B. Transformer in 2D Vision

The earliest transformer [25] was proposed for dealing with machine translation tasks. Later, many frameworks introduced the attention mechanism for processing visual tasks. Wu et al. [27] proposed a visual transformer, which is based on the role of marked image analysis in feature mapping. Furthermore, ViT [26] also made considerable achievements, which were completely based on self-attention as it did not have any convolution operators.

However, point clouds contain disordered and irregular data. A single point has no semantics, and it is difficult to mark a point as a patch sequence as can be done with an image. Therefore, we propose a neighbor embedding module to aggregate from the local neighborhood of a specific point to generate relevant semantic and local information.

C. Transformers in 3D Point Clouds

Our work is inspired by a previous idea: a point cloud, as a set with complete location attributes, fits with the set operator attributes of self-attention.

Recently, Zhao et al. [28] proposed the point transformer layer by applying vector self-attention. PCT [30] introduced point cloud transformation and achieved competitive results in point cloud classification and segmentation tasks. DPCT [41] aggregated both pointwise and channelwise self-attention models to semantically capture richer contextual dependencies from both the position and channel. Moreover, PTT [29] and PTTR [31] also applied transformers to cope with object tracking. Misra et al. introduced 3DETR [42], an end-to-end transformer-based object detection model. These methods show that transformers have broad applications in point clouds.

Unlike these methods, MPCT is implemented based on multiscale local neighborhoods with a residual network, which coincides with the design idea of [43]. Our work focuses on using a transformer and incorporates other strategies as supporting modules.

III. Method

We first introduce the general form of transformers and then propose our MPCT framework for point clouds. Given an input consisting of N points with xyz coordinates, a backbone network is designed to extract and learn the features of multiscale point clouds, and downstream tasks are designed to validate the output. Furthermore, we design some adaptive novel detail configurations for the proposed network.

A. Preliminaries For Transformers

Generally, self-attention operators can be divided into two types: scalar attention [25] and vector attention [44] operators.

Let $\mathcal{F}_{in} = \{\mathbf{f}_i\}_{i=1}^{N}$ be a set of feature vectors.

In scalar attention, the standard dot product attention layer can be expressed as follows:

$$Q, K, V = \mathcal{F}_{in} \cdot (W_q, W_k, W_v) \tag{1}$$

$$Q, K, V \in \mathbb{R}^{N\times D}, \mathcal{A} = \rho\left(QK^{\top}\right) V \tag{2}$$

where W_q, W_k and W_v are shared learnable transformations. Q, K, and V are the *query*, *key* and *value* matrices generated by the linear transformation of the input feature $\mathcal{F}_{in}$, respectively. ρ is a normalization function (e.g., as softmax), and $\mathcal{A}$ is the attention feature generated by the self-attention layer.

In vector attention, the attention calculation is different. In particular, attention can adjust certain feature channel:

$$\mathcal{A} = \rho(\gamma(\sigma(Q, K))) \otimes V \tag{3}$$

where σ is the relationship function between Q and K (e.g., subtraction or multiplication), γ is a mapping function (e.g., an MLP) that generates the attention vector, and $\otimes$ is the Hadamard product. Scalar and vector self-attention are essentially set operators, and the complexity of self-attention is $\mathcal{O}(N^2 D)$.

B. Proposed MPCT Architecture

We aim to leverage the self-attention network for point cloud analysis purposes. Note that the point transformer is an important feature aggregation operator for the entire network. We adopt point-enhanced geometric descriptor preprocessing without auxiliary branches. Based on this, we propose a powerful network architecture to analyze the potential information and capture the latent features in point clouds: the network is completely based on the point transformer layer, multiscale embedding, and pooling transformation. The proposed MPCT architecture is illustrated in Fig. 2.

Backbone: There are four stages in the feature transformation in MPCT. After the point information is enhanced, these stages operate on the point set that is gradually downsampled. The downsampling rates of the stage are $[1/2, 1/2, 1/2, 1/2]$, so the bases of the point sets generated in different stages are $[N/2, N/4, N/8, N/16]$, and the bases of the aggregation features are $[2D, 4D, 8D, 16D]$, where N is the number of input points, and D is the feature dimension. The adopted lightweight backbone structure, the number of stages, the downsampling rates, and the aggregate feature dimensions expressed in the figure can vary according to the given application. Overall, the feature embedding module maps the point-enhanced input features to the embedding space. Multiscale local embedding performs feature transformation operations on the point cloud with different degrees of sampling. The self-attention module learns refined attention features for the input features based on the local context. The output features are the residual sum of the input features and their corresponding features produced through the self-attention layer.

In contrast with PCT [30], which uses stacked attention modules, we map the given point cloud to different feature spaces after performing different sampling operations using a single block. The advantage of this massive simplification is that it avoids distractions across multiple channels. Therefore, communication between individual preprocessing blocks occurs in a single softmax layer, directly reflecting how the attention weight operator weights each block.

As a result, we attempt to illustrate the representation learning ability of transformers on two related tasks: point cloud classification and segmentation.

Classification: The details of using the classification network are shown in the upper part of Fig. 2 "Output". To recognize the input point cloud $\mathcal{P}$ as an object category (such as an airplane, a chair, etc.) for classification, we feed the multiscale fusion feature $\mathcal{F}_{out}$ (composed of attentional features generated during four stages connected after performing max pooling) to the classification decoder, which includes two cascaded LBRs. Each layer's dropout probability is 0.5, and then the point cloud is finally determined by the linear layer to predict the final classification score $\mathcal{C} \in \mathbb{R}^{N_c}$: the class with the largest score represents the class label of the point cloud.

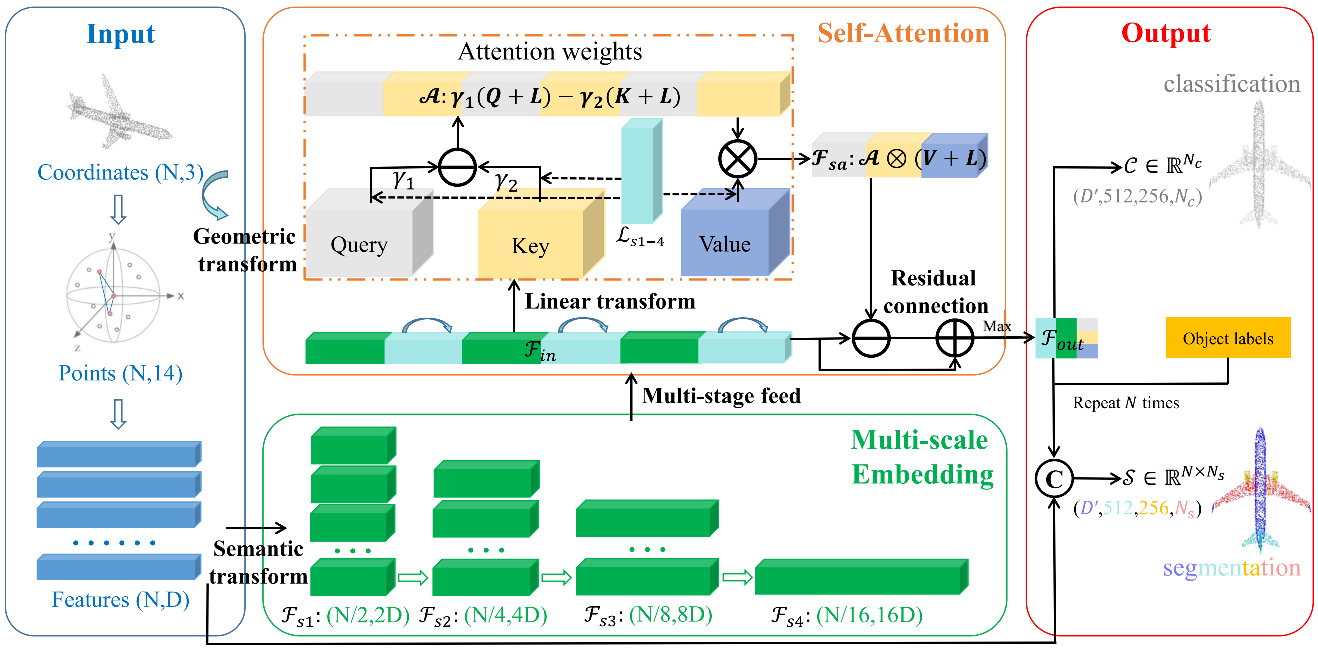


Fig. 2. Proposed MPCT architecture. MPCT aims to exploit rich point cloud features to build a powerful perception network, including four parts: position embedding, semantic embedding, multiscale embedding and self-attention modules. The input features (blue) are generated from the enhanced point information, and semantic features (green) are generated from the input features, and geometric features (cyan) are generated from the high-level geometric information of the initial points. The semantic features are decomposed into attention features via three linear transformations and fed forward along with the geometric features. $\oplus$, $\ominus$, $\otimes$ and $\mathcal{C}$ represent pairwise summation, subtraction, multiplication and connection operations performed via the feature channel, respectively.

Segmentation: The details of using the segmentation network are shown in the lower part of Fig. 2 "Output". The input point cloud is divided into N parts, which are represented by different points (e.g., a fuselage or a wing), and we need to predict a label for each point to represent a certain part of the object. First, we divide the single-scale feature $\mathcal{F}_{out_i} (i = 1, 2, 3, 4)$ of each stage with the pointwise feature $\mathcal{F}_{\mathcal{P}}$ of the input stage for feature propagation [15]. In addition, we encode the object labels, repeating this step N times after the MLP to correspond to the point labels. The segmentation network decoder is almost the same as that of the classification network, which is replaced with two MLPs, and dropout is only executed on the first MLP. Then, we predict the final point segmentation score $\mathcal{S} \in \mathbb{R}^{N \times N_s}$: the part label of a point is determined as the part with the largest score.

C. Information Enhancement Encoding

Positional encoding is crucial in attention, as it establishes local and long-range dependencies among heterogeneous input features. In point cloud networks, the geometric information of points is a natural candidate for position encoding.

Point Enhancement: Inspired by [45], we argue that the position information is very susceptible to the disturbance of local points. Hence, the smaller the value of K and the closer the relationship with the central point is better. Therefore, we propose a point enhancement method, which only needs to lift the two closest points to enhance the information of a certain point in the initial phase and convert the low-level descriptor to the high-level geometric descriptor.

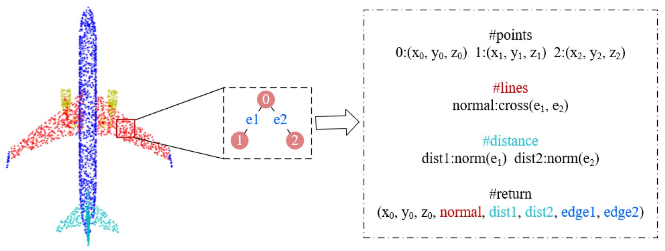

0
e1
e2
1
2
#points
0:(x0, y0, z0) 1:(x1, y1, z1) 2:(x2, y2, z2)
#lines
normal:cross(e1, e2)
#distance
dist1:norm(e1) dist2:norm(e2)
#return
(x0, y0, z0, normal, dist1, dist2, edge1, edge2)

Fig. 3. Point enhancement module uses the triangle relationship to explicitly enhance the low-level geometric relationships between points.

Specifically, we search for the two nearest neighborhoods of each point $p_i \in \mathbb{R}^3$ in the given point cloud, and denote them as $p_{j1}, p_{j2} \in \mathbb{R}^3$. By using them to form a triangle, we can estimate a geometric descriptor $\widetilde{p}_i$, which corresponds to p_i, as illustrated in Fig. 3.

Then we surpass the original point p_i by introducing a trainable and parameterized point $\widetilde{p}_i$. We argue that point encoding is more conducive to attention weighting and feature transformation. Therefore, the input features $\mathcal{F}_{in}$ are obtained from the trainable point encoding feature $\widetilde{p}_i$.

Neighborhood Enhancement: Usually, to encode the features of each point p_i, the geometry relationship is enhanced via the encoding of its K nearest points:

$$\theta\left(p_i, \forall p_{i_k}\right) = \left[\|RP(p_i, K) - \forall p_{i_k}\|, RP(p_i, K), \forall p_{i_k}, RP(p_i, K) - \forall p_{i_k}\right], \tag{4}$$

where θ is used to relate a point to its local neighborhoods, p_i is the i-th point in the point cloud, p_{i_k} is the k-th point in the local neighborhoods of p_i according to KNN, and $RP(f, K)$ is the operator for repeating a vector f K times to form a matrix. The position encoding module extracts higher-level information from the input coordinates to the local context $p_i \in \mathbb{R}^3 \rightarrow \forall l_{i_k} \in \mathbb{R}^{K\times 10}$.

Then, l_i' is locally aggregated, and the operation can be expressed as:

$$\forall l_{i_k}' = \mathcal{M}_1 \left(\forall l_{i_k} \mid k \in \mathcal{N}\left(p_i\right)\right), \forall l_{i_k}' \in \mathbb{R}^{K\times D}, \tag{5}$$

$$l_i' = MAX\left(\mathcal{M}_2(l_{i_k}') + l_{i_k}'\right), l_i' \in \mathbb{R}^D, \tag{6}$$

where $\mathcal{M}_1$ and $\mathcal{M}_2$ are MLP operators with a linear layer, a normalization layer and an activation layer by the provided local context, and MAX is the max pooling operator. D is the feature dimension, and its value may be different for different equations. When aggregating the local features, we also use the residual connectivity strategy and then connect the position encoding feature $\mathcal{L} = \{l_i'\}_{i=1}^N$, i.e., the geometric feature.

D. Point Cloud Transformer

As a recent study, PCT [30] uses *query* and *value* matrices to perform the dot product operation to measure the attention similarity, which requires squared complexity. Nonetheless, we believe that in point clouds, the dot product attention normalized by the softmax function [46] is not necessarily appropriate. Point Transformer [28] indicates that using elementwise subtraction to distinguish the correlations between points is more effective in terms of computational effectiveness.

In contrast, our point cloud transformer layer is based on pairwise attention with position encoding, and we adapt it more suitable for point clouds with the complexity of self-attention, which is $\mathcal{O}(ND)$, as shown in Fig. 4.

Specifically, we first use the *query* and *key* matrices to sum the location encoding matrix L, (the geometric feature is introduced in the next section) separately, and then all of the attention weights are calculated by pairwise subtraction:

$$\bar{\mathcal{A}} = \forall \bar{\alpha}_{i,j} = \gamma_1(Q + L) - \gamma_2(K + L), \tag{7}$$

where the subscripts i and j are the corresponding number and dimension, respectively. γ_1 and γ_2 are mapping functions composed of fully connected layers without bias. Then the attention weights are normalized to obtain $\mathcal{A} = \forall \alpha_{i,j}$:

$$\forall \alpha_{i,j} = Softmax\left(\forall \bar{\alpha}_{i,j}\right) = \forall \frac{\exp\left(\bar{\alpha}_{i,j}\right)}{\sum_k \exp\left(\bar{\alpha}_{i,k}\right)}, \tag{8}$$

where $Softmax$ serves to obtain a row-level normalization of the attention matrix $\mathcal{A} \in \mathbb{R}^{N\times D}$. In other words, we normalize each point in the input point cloud *w.r.t* all other points to obtain a weighted aggregation of the contextual information.

The output self-attention feature $\mathcal{F}_{sa}$ is the weighted sum of the *value* matrix obtained with the position encoding information using the corresponding attention weights:

$$\mathcal{F}_{sa} = \mathcal{A} \otimes (V + L). \tag{9}$$

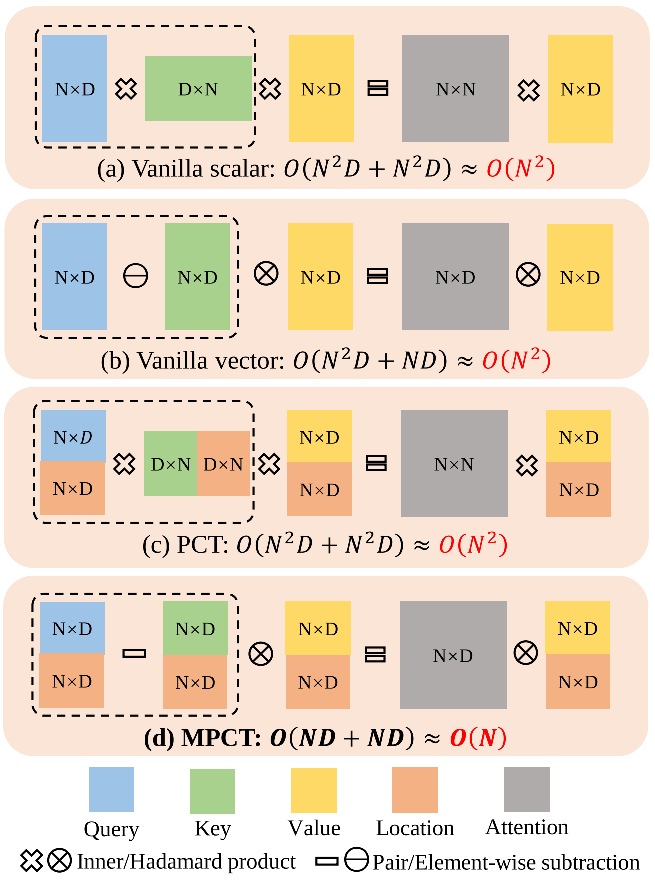

N×D
D×N
N×N
(a) Vanilla scalar: $O(N^2D + N^2D) \approx O(N^2)$
N×D
(b) Vanilla vector: $O(N^2D + ND) \approx O(N^2)$
(c) PCT: $O(N^2D + N^2D) \approx O(N^2)$
(d) MPCT: $O(ND + ND) \approx O(N)$
Query
Key
Value
Location
Attention
Inner/Hadamard product
Pair/Element-wise subtraction

Fig. 4. Overview of attention for scalar attention [25], vector attention [44], PCT [30], and our MPCT. The number of points is N, and the feature dimension is D, $D \ll N$.

Since the *query*, *key* and *value* matrices are determined by the shared linear transformation of the input feature $\mathcal{F}_{in}$, they are all permutation-invariant. Moreover, the softmax and weighted sum operators are independent of the order. Therefore, the entire self-attention process is arranged invariantly, which is suitable for the irregular point cloud domain.

Finally, the self-attention feature $\mathcal{F}_{sa}$ and the input feature $\mathcal{F}_{in}$ are further used for residual connection purposes, and the output feature $\mathcal{F}_{out}$ is provided for the entire self-attention layer after the Linear, Batchnorm, and ReLU (LBR) network:

$$\mathcal{F}_{out} = \mathcal{F}_{in} + LBR(\mathcal{F}_{in} - \mathcal{F}_{sa}). \tag{10}$$

E. Multiscale Feature Fusion

Most point-based learning methods are designed to efficiently extract global features. Nonetheless, recent work has fully demonstrated that local neighborhood information is essential for point cloud feature analysis. We refer to the ideas of PointNet++ [15] and DGCNN [16] and further design a multiscale aggregation strategy that integrates different neighborhoods to enhance the network's local feature extraction capability.

The local embedding module includes four sampling and grouping (SG) layers and four LBR layers, which are used to extract feature information at four scales. We hope to use four cascaded SG layers to gradually expand the receptive field during feature aggregation so that the neural network can see more refined features. The SG layers use the Euclidean distance measure to search for each point grouped by K-NN during point cloud sampling to aggregate the features derived from local neighborhoods. To facilitate the distinction, we directly refer to the features obtained via the aggregation of the point features as semantic features, whose search and connection strategies are the same as those of the position encoding features.

Corresponding to the geometric features above, $\theta(f_i, \forall f_{i_k}) = [RP(f_i, K), RP(f_i, K) - \forall f_{i_k}]$ is typically used to denote the local semantic context in the feature space. Nevertheless, $\theta(f_i, \forall f_{i_k})$ may not be sufficient for representing the neighborhood because of two reasons. 1) Due to the sparse and irregular geometric structures of point clouds, their generalization capabilities in high-dimensional feature spaces may be weakened, and 2) since point cloud neighbors are not unique to each other in the representation of closed regions, information redundancy may occur. To address these issues and enhance the generalization ability of the features, we transform the local points into a normal distribution while maintaining their original semantics:

$$\forall f_{i_k} = \alpha \times \frac{\forall f_{i_k} - f_i}{\sigma + \epsilon} + \beta, \quad \sigma = \sqrt{\frac{1}{K} \sum\nolimits_{k=1}^{K} (f_{i_k} - f_i)^2}, \tag{11}$$

where $\alpha \in \mathbb{R}^D$ and $\beta \in \mathbb{R}^D$ are learnable parameters and $\epsilon = 1e-8$ is a small number for numerical stability. liujiamingshi-daSB Then, we use farthest point sampling (FPS) [15] to sample $\mathcal{P}$ to $\mathcal{P}_{s1}$ for each sampling point $p \in \mathcal{P}_{s1}$, letting $\mathcal{N}_{\mathcal{P}}(p)$ be p-th nearest neighborhoods in $\mathcal{P}$. Through the same processing approach conducted on the basis of $\mathcal{P}_{s1}$, we suggest that the four sampling rates and neighborhood numbers are equal, with values of 0.5 and 16, respectively. Next, we calculate the corresponding local features:

$$\mathcal{F}_{s1}(p) = [RP(f_{s1}(p), K), \forall f_{i_k}], \mathcal{F}_{s1}(p) \in \mathbb{R}^{K \times 2D}, \tag{12}$$

$$\mathcal{F}_{s1}(p) = MAX(\mathcal{M}(\mathcal{F}_{s1}(p)) + \mathcal{F}_{s1}(p)), \mathcal{F}_{s1}(p) \in \mathbb{R}^{2D}, \tag{13}$$

where $f_{s1}(p)$ is the feature of point p in sampled point cloud $\mathcal{P}_{s1}$, and $\mathcal{F}_{s1}(p)$ is the feature of sampling point p after aggregating the neighbor points, i.e., the input feature of the self-attention network module.

Next, we connect the point features of the sampled $\mathcal{P}_{s1}$ and regard them as the research objects for generating $\mathcal{P}_{s2}$, $\mathcal{P}_{s3}$, and $\mathcal{P}_{s4}$. After executing a series of transformations, we obtain the feature representing the multiscale local features:

$$\mathcal{F}_{si} = \sum_{pi \in \mathcal{P}_{si}} \mathcal{F}_{si}(pi) \ \ s.t. \ \ i = 1, 2, 3, 4. \tag{14}$$

The numbers of points and feature dimensions at each sampling scale are different and inversely proportional, and the generated multiscale features can separately construct local dependencies between the point clouds. Nonetheless, simply connecting the point clouds at this time may ignore the remote dependencies between them, so we feed these features into the self-attention network, which can be expressed as follows:

$$\mathcal{A}_{s1} = \mathcal{A}(\mathcal{F}_{s1}, \mathcal{L}_{s1}), \mathcal{A}_{s1} \in \mathbb{R}^{N \times D}, \tag{15}$$

where $\mathcal{A}_{s1}$ and $\mathcal{L}_{s1}$ are the attentional feature via the self-attention function and the geometric feature mentioned before, respectively. Similarly, we can obtain $\mathcal{A}_{s2}$, $\mathcal{A}_{s3}$ and $\mathcal{A}_{s4}$.

Eventually, we use the generated attentional features as conditions for point cloud understanding to evaluate the point cloud classification and segmentation tasks.

IV. Experiments

In this section, we evaluate the validity of the proposed MPCT for point cloud downstream tasks on five datasets, i.e., ModelNet40 [47], ScanObjectNN [48], ShapeNetPart [49], S3DIS [50], and ScanNet-V2 [51], and we conduct multiple comprehensive comparisons with other methods.

A. Implementation Details

In general, we train MPCT using the cross-entropy loss with label smoothing [52] and evaluate MPCT using an entire scene as its input.

For shape classification, we use SGD [53] optimization for 300 epochs with a momentum of 0.9 and a weight decay of 0.0001, and cosine annealing reduces the learning rate from 0.1 to 0.001. For part and semantic segmentation, we use AdamW [54] optimization for 200 epochs, an initial learning rate $lr = 0.002$, and a weight decay of 10–4, with cosine decay. The training batch size is 32, and the test batch size is 16.

In addition, the training data are enhanced with a random translation range of [−0.2,0.2] and a random scale ratio of [2/3,3/2], and strategies such as point resampling, additional height, random sampling, etc., are also included [55]. We use PyTorch to implement the project, and all experiments are performed on two GeForce RTX 3090 GPUs.

B. Classification Results

The synthetic ModelNet40 dataset has 40 object categories and contains 12,311 CAD models. They are divided into 9,843 models for training and 2,468 models for testing.

Table I shows the overall and average classification accuracies achieved by MPCT on ModelNet40. Specifically, we achieve an overall accuracy of 94.2% and an average classification accuracy of 92.0%, which exceeds the results obtained with other similar inputs overall. Notably, it can be seen that our method performs better than the other methods that use more input points or features.

The ScanObjectNN real-world dataset has 15 categories with 2902 unique object instances. It contains more than 15,000 objects, and it is actually more challenging due to the complexity of its backgrounds and the locality of the objects. We use its most challenging version, *PBT50RS*, with noisy background points for experiments. We use the mean class accuracy (mAcc) for each category and the overall accuracy (OA) calculated across all categories as evaluation metrics. To facilitate the comparison experiments, we do not use a voting strategy.

TABLE I
SHAPE CLASSIFICATION RESULTS (%) OBTAINED ON MODELNET40

Methods		Input	mAcc	OA
Point Methods	PointNet [11] (2017)	P	86.2	89.2
	PointNet++ [12] (2017)	P	-	91.9
	PointWeb [53] (2019)	P	89.4	92.3
	PointMLP [40] (2022)	P	91.3	94.1
Convolution Methods	PointConv [18] (2019)	P+N	-	92.5
	KPConv [14] (2019)	P	-	92.9
	PAConv [54] (2021)	P	-	93.6
	SC-CNN [55] (2022)	P	-	93.8
Graph Methods	RGCNN [56] (2018)	P+N	87.3	90.5
	DGCNN [13] (2019)	P	90.2	92.2
	AdaptConv [57] (2021)	P	90.7	93.4
	AGNet [58] (2022)	P	90.7	93.4
Others Methods	VoxNet [9] (2015)	voxel	83.0	85.9
	MVCNN [7] (2015)	image	-	90.1
	CurveNet [59] (2021)	P	-	93.8
	EllipsoidNet [60] (2022)	P	91.4	92.7
Transformer Methods	PCT [27] (2021)	P	-	93.2
	PT [25] (2021)	P	90.6	93.7
	CSANet [61] (2022)	P	89.9	92.8
	PatchFormer [62] (2022)	P	-	93.5
	MPCT (ours)	P	**92.0**	**94.2**

P = point, N = normal. All experimental results except MPCT are from the corresponding cited articles, as below.

Table II shows the results yielded by MPCT on the ScanObjectNN dataset, which includes actual scan data containing real-world objects. Our method's overall accuracy rate of 84.9% and average accuracy rate of 82.9% are significantly higher than all other results and are first in the official rankings, and the *PBT50RS* variant used in the experiment is still the most challenging version. Notably, we are in a leading position in many of the 15 categories, and the accuracy of normal object prediction is directly proportional to the number of models.

C. Segmentation Results

The ShapeNetPart dataset is an object-oriented dataset with 16,880 object point clouds belonging to 16 categories, and each point is labeled as one of 50 parts; its models are divided into 14,006 models for training and 2,784 models for testing.

Table III shows the results produced by MPCT on ShapeNetPart, which achieves the best results, with improvements of 3.0% and 1.5% over PointNet and the DGCNN in terms of their overall IOU results, respectively. Fig. 5 shows further segmentation examples provided by PointNet, DGCNN, and MPCT.

The S3DIS dataset is a dataset of indoor scenes for semantic segmentation of point clouds, which contains 6 areas and 271 rooms, and each point in the dataset is divided into 13 categories. We use mAcc and intersection-over-union (IoU) as evaluation metrics. The IoU of a shape is calculated from the average of the IoUs of all parts of the shape, and mIoU is the mean of the IoUs of all tested shapes.

Table IV shows the results obtained by MPCT on the S3DIS dataset. Following PointNet++'s protocol, we evaluate Area 5. Our MPCT achieves 74.6%/68.6% mAcc/mIoU. The results are close to those of the latest PT presented by [28] and are multiple percentage points higher for each semantic label than those of previous works. Fig. 6 shows the MPCT predictions, and we can see that the predictions are close to the real situation.

D. Object Detection Results

To verify the generalization ability of MPCT, we extend its application to 3D object detection. Specifically, we use 3DETR [42] as a baseline and port MPCT as a general component to the transformer encoder in the backbone of 3DETR. Note that we change the configuration of MPCT and embed it in 3DETR as normal, while leaving the other settings unchanged.

We report our results obtained on ScanNet-V2 [51], which has 1021 training samples, 312 validation samples, 100 testing samples, and axis-aligned bounding box labels for 18 object categories. Follow the experimental protocol in [70]: we report the detection performance achieved on the validation set using mean average precision (mAP) attained at two IoU thresholds (0.25 and 0.5, denoted as AP_{25} and AP_{50}, respectively).

As shown in Table V, the revised 3DETR with our MPCT yields a significant improvement (+0.9AP_{25} and +2.8AP_{50}) over the vanilla 3DETR. This result suggests that the network's encoder can be facilitated to learn richer representations for input point clouds under comprehensive transform processing.

E. Computational Analysis

We consider the computational efficiency levels of MPCT and several other methods by comparing their numbers of required parameters (Params), numbers of floating point operations (FLOPs), and inference time in Table VI. Among them, the single-scale PointNet++ has the lowest memory requirement of 1.48 M parameters, and PointNet requires the lowest processor load of 0.45 G FLOPs while providing less accurate results. Overall, MPCT has the best performance with moderate computational and memory requirements.

F. Ablation Studies

We perform several ablation experiments in this section, choosing alternatives for the geometric point descriptor, the attention mechanisms, the positional encoding, and the residual structure to understand the value of our constructions. All experimental steps and evaluation metrics are implemented in the same setup as that used for the classification experiments.

Number of Local Neighborhoods: First, we study the number of neighbors k, which is used to determine the local neighborhoods around each point. The results are shown in Table VII. The best performance is obtained when k is set to 16. When the number of neighborhoods is small ($k = 8$ or $k = 12$), the model may not have enough context to make predictions. When the number of neighborhoods is large ($k = 24$ or $k = 32$), each self-attention layer provides a large number of data points, many of which may be more distant and less relevant. In addition, the artificial weak noise added during training is intended to enhance the network's learning ability, but it instead reduces the model's accuracy when the number of neighbors is large.

TABLE II
SHAPE CLASSIFICATION RESULTS (%) OBTAINED ON THE SCANOBJECTNN DATASET

Methods	OA	mAcc	bag	bin	box	cabinet	chair	desk	display	door	shelf	table	bed	pillow	sink	sofa	toilet
#shapes	-	-	298	794	406	1344	1585	592	678	892	1084	922	564	405	469	1058	325
3DmFV [63]	63.0	58.1	39.8	62.8	15.0	65.1	84.4	36.0	62.3	85.2	60.6	66.7	51.8	61.9	46.7	72.4	61.2
PointNet [11]	68.2	63.4	36.1	69.8	10.5	62.6	89.0	50.0	73.0	**93.8**	72.6	67.8	61.8	67.6	64.2	76.7	55.3
PointNet++ [12]	77.9	75.4	49.4	84.4	31.6	77.4	91.3	**74.0**	79.4	85.2	72.6	72.6	75.5	**81.0**	**80.8**	90.5	85.9
DGCNN [13]	78.1	73.6	49.4	82.4	33.1	**83.9**	91.8	63.3	77.0	89.0	79.3	**77.4**	64.5	77.1	75.0	91.4	69.4
PointCNN [64]	78.5	75.1	57.8	82.9	33.1	83.6	92.6	65.3	78.4	84.8	84.2	67.4	80.0	80.0	72.5	91.9	71.8
DRNet [65]	80.3	78.0	**66.3**	81.9	49.6	76.3	91.0	65.3	**92.2**	91.4	83.8	71.5	79.1	75.2	75.8	91.9	78.8
GBNet [42]	80.5	77.8	59.0	84.4	44.4	78.2	92.1	66.0	91.2	91.0	**86.7**	70.4	82.7	78.1	72.5	92.4	77.6
MPCT (ours)	**84.9**	**82.9**	66.2	**88.9**	**59.6**	83.8	**93.8**	69.3	91.7	92.5	85.6	73.2	**87.4**	76.2	80.2	**92.8**	**88.2**

TABLE III
PART SEGMENTATION RESULTS (%) OBTAINED ON THE SHAPENETPART DATASET

Methods	overall mIou	air-plane	bag	cap	car	chair	ear-phone	guitar	knife	lamp	laptop	motor-bike	mug	pistol	rocket	skate-board	table
#shapes	-	2690	76	55	898	3758	69	787	392	1547	451	202	184	283	66	152	5271
PointNet [11]	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [12]	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
DGCNN [13]	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
PointCNN [64]	86.1	84.1	**86.5**	86.0	80.8	90.6	79.7	92.3	88.4	85.3	96.1	77.2	95.2	84.2	64.2	80.0	83.0
DRNet [65]	86.4	84.3	85.0	88.3	79.5	91.2	79.3	91.8	**89.0**	85.2	95.7	72.2	94.2	82.0	60.6	76.8	84.2
PointMLP [40]	86.1	83.5	83.4	87.5	80.5	90.3	78.2	92.2	88.1	82.6	96.2	**77.5**	**95.8**	**85.4**	**64.6**	**83.3**	**84.3**
PCT [27]	86.4	**85.0**	82.4	89.0	**81.2**	91.9	71.5	91.3	88.1	86.3	95.8	64.6	95.8	83.6	62.2	77.6	83.7
SC-CNN [55]	86.4	83.6	85.4	88.5	80.4	91.6	**81.9**	91.8	88.5	85.6	96.1	75.5	94.8	83.5	59.7	77.4	83.6
MPCT (ours)	**86.7**	84.9	85.7	**89.3**	80.9	**92.2**	78.6	**92.6**	88.4	**86.6**	**96.5**	73.6	95.7	83.5	63.7	80.9	83.6

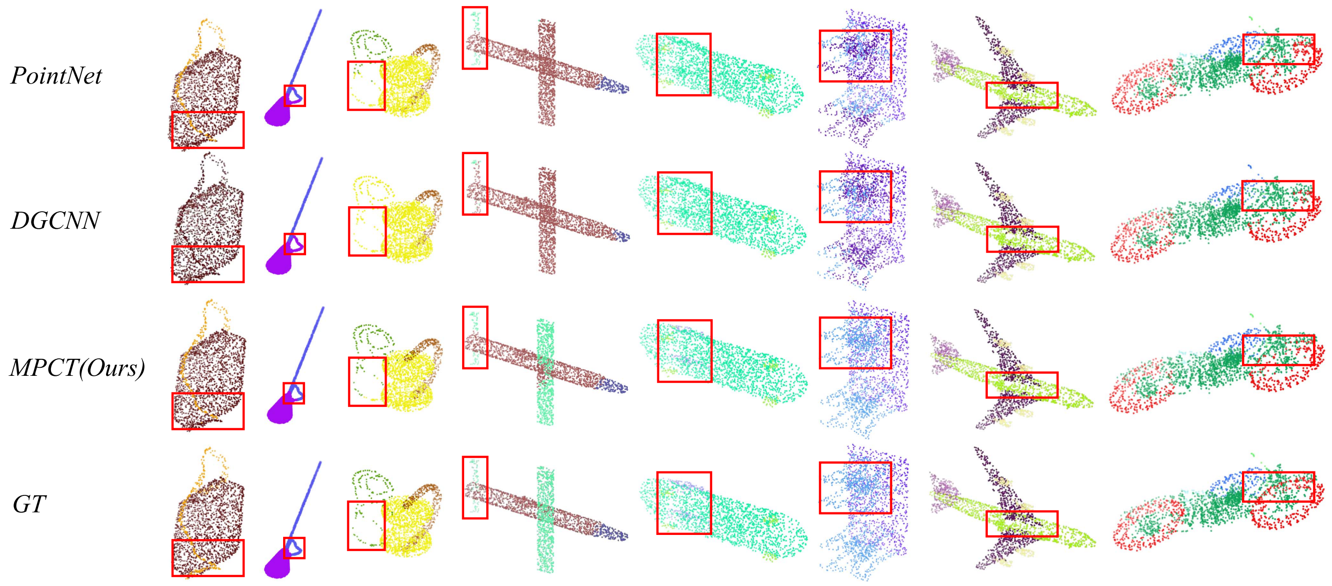


Fig. 5. Visualization of the part segmentation results obtained on the ShapeNetPart dataset. We highlight the different predictions in red boxes.

Geometric Point Descriptor: We next confirm that geometric point descriptors are explicitly formed in 3D space and include four low-level geometric descriptors: coordinates, edges, normals, and edge lengths. The results are shown in Table VIII. As an essential condition, the coordinates of a point can intuitively represent global information. In addition, the edge of a point can imply more local information by referring to the relative positions of its neighbors, and the length of the edge can partially estimate the density distribution of the neighbors of the corresponding point. Furthermore, we use the cross product of the edge vectors to compute the point normals to enhance the robustness of the descriptor to possible deformations such as scaling, translation, rotation, etc. However, combining all this information is not always optimal because it may contain redundant information. For example, redundant point information, i.e., p_{j1} and p_{j2}, which does not help to describe the low-level geometric features of point p_i.

TABLE IV
SEMANTIC SEGMENTATION RESULTS (%) OBTAINED ON THE S3DIS DATASET, EVALUATED ON AREA 5

Methods	OA	mAcc	mIoU
PointNet [11]	-	49.0	41.1
PointNet++ [12]	-	63.5	57.2
PointCNN [64]	85.9	63.9	57.3
PAT [66]	-	70.8	60.1
PCT [27]	-	67.6	61.3
PAConv [54]	-	-	66.3
KPConv [14]	-	72.8	67.1
MPCT (ours)	**89.3**	**74.6**	**68.6**

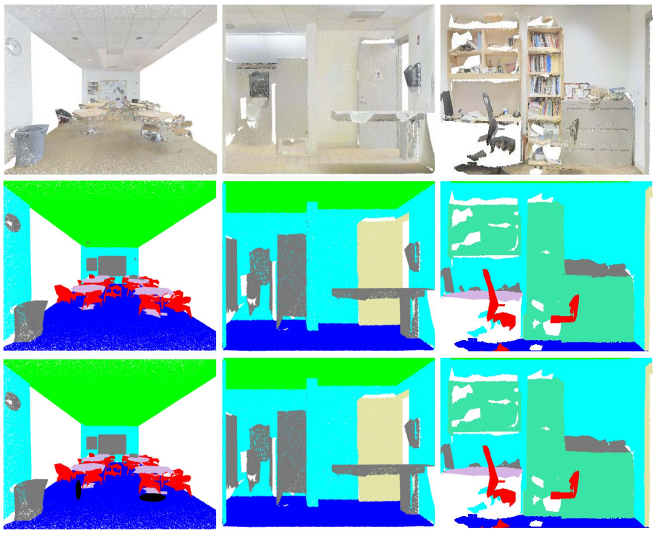


Fig. 6. Visualization of semantic segmentation results obtained on the S3DIS dataset. Top: input, middle: GT, bottom: MPCT (ours).

TABLE V
OBJECT DETECTION RESULTS (%) ON THE SCANNET-V2 DATASET

Methods	AP_{25}	AP_{50}
3DETR [39]	62.7	37.5
+ MPCT	**63.6(+0.9)**	**40.3(+2.8)**

TABLE VI
COMPUTATIONAL RESOURCE REQUIREMENTS AND INFERENCE TIMES OF DIFFERENT METHODS, TESTED ON A GEFORCE RTX 3090 GPU

Methods	#Params	#FLOPs	Inference time
PointNet [11]	3.47M	**0.45G**	**16.6ms**
PointNet++ [12]	**1.48M**	1.68G	163.2ms
DGCNN [64]	1.81M	2.43G	27.2ms
KPConv [14]	15.2M	-	210.3ms
GBNet [42]	8.39M	-	32.2ms
PointMLP [40]	12.6M	-	31.8ms
PCT [27]	2.88M	2.32G	35.6ms
EllipsoidNet [60]	37.5M	-	96.4ms
MPCT (ours)	2.64M	2.16G	25.5ms

TABLE VII
ABLATION STUDY: NUMBER OF LOCAL NEIGHBORHOODS k

k	Input	#Points	OA	mAcc
[8,8,8,8]	P	1k	93.1	90.8
[12,12,12,12]	P	1k	93.8	91.6
[16,16,16,16]	P	1k	**94.2**	**92.0**
[24,24,24,24]	P	1k	94.0	91.4
[32,32,32,32]	P	1k	93.2	90.6

TABLE VIII
ABLATION STUDY: GEOMETRIC POINT DESCRIPTOR

Geometric point descriptor	length	OA	mAcc
$\widetilde{p_i} = (p_i)$	3	93.5	91.1
$\widetilde{p_i} = (p_i, n_i, d_1, d_2)$	8	93.6	91.3
$\widetilde{p_i} = (p_i, d_1, d_2, e_1, e_2)$	11	93.6	91.4
$\widetilde{p_i} = (p_i, n_i, e_1, e_2)$	12	93.7	91.3
$\widetilde{p_i} = (p_i, n_i, d_1, d_2, e_1, e_2)$	14	**94.2**	**92.0**
$\widetilde{p_i} = (p_i, n_i, p_{j1}, p_{j2}, e_1, e_1)$	18	93.3	90.9

TABLE IX
ABLATION STUDY: THE FORMS OF THE SELF-ATTENTION OPERATORS

Operator	Input	#Points	OA	mAcc
MLP	P	1k	92.7	88.8
Scalar attention	P	1k	93.1	90.4
Vector attention	P	1k	93.3	90.8
PCT attention	P	1k	93.5	91.2
Our attention	P	1k	**94.2**	**92.0**

Attention Operator: Next, we study the types of self-attention used in the point transformation layer. The results are shown in Table IX. We examine five cases: MLP is an inattentive baseline that replaces the point transformation layer with a pointwise MLP. *Scalar attention* replaces the vector attention with scalar attention, as in (2). *Vector attention* is the pre-improved version, as shown in (3). We use (7) as our self-attention operator. Scalar attention is more expressive than the no-attention baseline but not as good as vector attention. The performance gap between our attention mechanism and PCT attention is also pronounced: the OA/mAcc values are 94.2%/92.0% vs. 93.5%/91.2%. Our attention operator is more expressive on the basis of vector attention and is very beneficial for 3D data processing.

Position Encoding: We then study the choice of the position encoding $\mathcal{L}$, as shown in Table X. Our relative position encoding implemented according to (4) significantly improves the performance. We argue that the coordinates of a point can intuitively represent global information, while its edges can imply more local information by referring to its neighbors, and the length of an edge can estimate the density distribution of the neighbors of the corresponding point.

Residual Structure: Finally, we perform an ablation study on the residual connection used in Equations (10) and (13).

TABLE X
ABLATION STUDY: THE POSITION ENCODING AND RESIDUAL STRUCTURE

Pos. encoding			Res. structure		OA	mAcc
none	absolute	relative	without	with		
✓			✓		91.0	87.9
✓				✓	92.9	89.8
	✓		✓		92.8	89.4
	✓			✓	93.6	91.5
		✓	✓		93.3	91.3
		✓		✓	**94.2**	**92.0**

The results are shown in Table X. We can see that the performance achieved without the residual structure on ModelNet40 is 93.3%/91.3% in terms of the OA/mAcc metrics, which is much lower than the performance attained using the residual structure (0.9%/0.7%). This indicates that even a simple residual structure is essential in this case.

G. Limitations

Our MPCT and the previous approaches, such as PT [28], PCT [30], and PointMLP [43], contain similar parts, including a transformer, a position encoding module, and a residual network. However, we make integrative improvements.

For instance, we adopt a pairwise mechanism to calculate attention similarity with a lower linear $\mathcal{O}(ND)$ complexity. Our work is novel in that we are the first to systematically apply a residual network to high-level geometric and semantic features extracted from point clouds and demonstrate its effectiveness in a simple and efficient manner. Additionally, our results validate the proposed method through the use of multiscale point clouds. The differences between our approach and the main reference works are as follows.

- In PT [28], only single-scale features are used. Even though PT obtains point clouds with different resolutions through a series of downsampling operations and then aggregates them through corresponding upsampling operations, information is still inevitably lost at different stages of network. In contrast, our MPCT with a residual network can analyze and generalize point cloud features at different scales to effectively solve this problem.
- PCT [30] (see Fig. 1) only uses a single-scale embedding and does not consider position encoding, using a more complex and computationally intensive attention mechanism (refer to Fig. 4(c)). We achieve advantages in terms of both efficiency and performance by applying a pairwise subtractive attention mechanism (see Fig. 4(d)).
- PointMLP [43] does not use a transformer and provides no obvious judgments concerning the geometric and semantic features of the given point cloud. Inspired by this, we embed an improved residual network in our novel point cloud transformer, and achieve an effective improvement.

Looking back at the proposed MPCT itself, even if the cost of self-attention is reduced by utilizing subtractive pairwise attention, it has been proven to be effective for point cloud understanding. However, MPCT only performs the corresponding operation in the local domain and does not consider setting the attention mechanism from the global position of the point cloud. Furthermore, the performance of switching this attention to complex point cloud scenes is unknown. This is a new challenge that we need to face in future work.

V. CONCLUSION

In this article, we propose a transformer-based architecture that is applied to point clouds. As point clouds are essentially geometric positions in metric space, the core self-attention operator of the transformer network fits well with position operations. We further improve the self-attention network and generative features and introduce the residual network to maintain continuity when processing the various stages of the given point cloud. Extensive experiments show that MPCT has a good feature learning capability and achieves advanced performance on several benchmarks, especially shape classification and semantic segmentation. Note that our research is more comprehensive and in-depth than previous pioneering works. In the future, we hope that our work will inspire further studies on the characteristics of point cloud transformers.

ACKNOWLEDGMENT

We gratefully acknowledge the support of MindSpore, CANN (Compute Architecture for Neural Networks) and Ascend AI Processor used for this research.

REFERENCES

[1] T. Weng, J. Xiao, F. Yan, and H. Jiang, “Context-aware 3D point cloud semantic segmentation with plane guidance,” *IEEE Trans. Multimedia*, early access, Oct. 10, 2022, doi: 10.1109/TMM.2022.3212914.
[2] Y. Wu et al., “Multi-view point cloud registration based on evolutionary multitasking with bi-channel knowledge sharing mechanism,” *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 2, pp. 357–374, Apr. 2023.
[3] C. Sun, Z. Zheng, X. Wang, M. Xu, and Y. Yang, “Self-supervised point cloud representation learning via separating mixed shapes,” *IEEE Trans. Multimedia*, early access, Sep. 14, 2022, doi: 10.1109/TMM.2022.3206664.
[4] Y. Wu et al., “Self-supervised intra-modal and cross-modal contrastive learning for point cloud understanding,” *IEEE Trans. Multimedia*, early access, Jun. 09, 2023, doi: 10.1109/TMM.2023.3284591.
[5] Y. Wu et al., “RORNet: Partial-to-partial registration network with reliable overlapping representations,” *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 30, 2023, doi: 10.1109/TNNLS.2023.3286943.
[6] Z. Zhang, J. Chen, X. Xu, C. Liu, and Y. Han, “Hawk-eye-inspired perception algorithm of stereo vision for obtaining orchard 3D point cloud navigation map,” *CAAI Trans. Intell. Technol.*, to be published.
[7] H. Wang, D. Huang, and Y. Wang, “GridNet: efficiently learning deep hierarchical representation for 3D point cloud understanding,” *Front. Comput. Sci.*, vol. 16, no. 1, 2022, Art. no. 161301.
[8] J. Liu et al., “Instance-guided point cloud single object tracking with inception transformer,” *IEEE Trans. Instrum. Meas.*, to be published.
[9] Y. Guo et al., “Deep learning for 3D point clouds: A survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.
[10] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, “Multi-view convolutional neural networks for 3D shape recognition,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 945–953.
[11] A. Kanezaki, Y. Matsushita, and Y. Nishida, “RotationNet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 5010–5019.
[12] D. Maturana and S. Scherer, “VoxNet: A 3D convolutional neural network for real-time object recognition,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.
[13] C. Lv, W. Lin, and B. Zhao, “Voxel structure-based mesh reconstruction from a 3D point cloud,” *IEEE Trans. Multimedia*, vol. 24, pp. 1815–1829, 2022.

[14] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.
[15] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.
[16] Y. Wang et al., "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.
[17] H. Thomas et al., "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6410–6419.
[18] Y. Wu et al., "Evolutionary multiform optimization with two-stage bidirectional knowledge transfer strategy for point cloud registration," *IEEE Trans. Evol. Comput.*, early access, Oct. 19, 2022, doi: 10.1109/TEVC.2022.3215743.
[19] Y. You et al., "PRIN/SPRIN: On extracting point-wise rotation invariant features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 9489–9502, Dec. 2022.
[20] Y. Wu et al., "Evolutionary multitasking with solution space cutting for point cloud registration," *IEEE Trans. Emerg. Topics Comput. Intell.*, early access, Jul. 12, 2023, doi: 10.1109/TETCI.2023.3290009.
[21] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9613–9622.
[22] Y. Wu et al., "INENet: Inliers estimation network with similarity learning for partial overlapping registration," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 3, pp. 1413–1426, Mar. 2023.
[23] X. Lin, K. Chen, and K. Jia, "Object point cloud classification via polyconvolutional architecture search," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 807–815.
[24] Y. Wu et al., "Commonality autoencoder: Learning common features for change detection from heterogeneous images," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 9, pp. 4257–4270, Sep. 2022, doi: 10.1109/TNNLS.2021.3056238.
[25] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[26] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.
[27] B. Wu et al., "Visual transformers: Token-based image representation and processing for computer vision," 2020, *arXiv:2006.03677*.
[28] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point transformer," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16239–16248.
[29] J. Shan, S. Zhou, Z. Fang, and Y. Cui, "PTT: Point-track-transformer module for 3D single object tracking in point clouds," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 1310–1316.
[30] M.-H. Guo et al., "PCT: Point cloud transformer," *Comput. Vis. Media*, vol. 7, no. 2, pp. 187–199, 2021.
[31] C. Zhou et al., "PTTR: Relational 3D point cloud object tracking with transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 8521–8530.
[32] Y. Wu et al., "SACF-Net: Skip-attention based correspondence filtering network for point cloud registration," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 8, pp. 3585–3595, Aug. 2023.
[33] Y. Wu et al., "PANet: A point-attention based multi-scale feature fusion network for point cloud registration," *IEEE Trans. Instrum. Meas.*, vol. 72, 2023, Art. no. 2512913.
[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[35] N. J. Mitra and A. Nguyen, "Estimating surface normals in noisy point cloud data," in *Proc. 19th Annu. Symp. Comput. Geometry*, 2003, pp. 322–328.
[36] Q. Mérigot, M. Ovsjanikov, and L. J. Guibas, "Voronoi-based curvature and feature estimation from point clouds," *IEEE Trans. Vis. Comput. Graph.*, vol. 17, no. 6, pp. 743–756, Jun. 2011.
[37] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8887–8896.
[38] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11105–11114.
[39] Y. Zhang, Q. Yang, and Y. Xu, "MS-GraphSim: Inferring point cloud quality via multiscale graph similarity," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 1230–1238.
[40] W.-X. Tao, G.-Y. Jiang, Z.-D. Jiang, and M. Yu, "Point cloud projection and multi-scale feature fusion network based blind quality assessment for colored point clouds," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 5266–5272.
[41] X.-F. Han, Y.-F. Jin, H.-X. Cheng, and G.-Q. Xiao, "Dual transformer for point cloud analysis," *IEEE Trans. Multimedia*, early access, Aug. 11, 2022, doi: 10.1109/TMM.2022.3198318.
[42] I. Misra, R. Girdhar, and A. Joulin, "An end-to-end transformer model for 3D object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 2886–2897.
[43] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual MLP framework," in *Proc. Int. Conf. Learn. Representations*, 2022.
[44] H. Zhao, J. Jia, and V. Koltun, "Exploring self-attention for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10073–10082.
[45] S. Qiu, S. Anwar, and N. Barnes, "Geometric back-projection network for point cloud classification," *IEEE Trans. Multimedia*, vol. 24, pp. 1943–1955, 2022.
[46] Q. Zhen et al., "CosFormer: Rethinking softmax in attention," in *Proc. Int. Conf. Learn. Representations*, 2022.
[47] Z. Wu et al., "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1912–1920.
[48] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1588–1597.
[49] L. Yi et al., "A scalable active framework for region annotation in 3D shape collections," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 1–12, 2016.
[50] I. Armeni et al., "3D semantic parsing of large-scale indoor spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1534–1543.
[51] A. Dai et al., "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.
[52] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.
[53] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 2818–2826.
[54] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2017, *arXiv:1711.05101*.
[55] G. Qian et al., "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 23192–23204.
[56] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "PointWeb: Enhancing local neighborhood features for point cloud processing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5560–5568.
[57] M. Xu, R. Ding, H. Zhao, and X. Qi, "PAConv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3172–3181.
[58] C. Wang et al., "Learning discriminative features by covering local geometric space for point cloud analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5703215.
[59] G. Te, W. Hu, A. Zheng, and Z. Guo, "RGCNN: Regularized graph CNN for point cloud segmentation," in *Proc. 26th ACM Int. Conf. Multimedia*, 2018, pp. 746–754.
[60] H. Zhou et al., "Adaptive graph convolution for point cloud analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 4945–4954.
[61] W. Jing et al., "AGNet: An attention-based graph network for point cloud classification and segmentation," *Remote Sens.*, vol. 14, no. 4, 2022, Art. no. 1036.
[62] T. Xiang, C. Zhang, Y. Song, J. Yu, and W. Cai, "Walk in the cloud: Learning curves for point clouds shape analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 895–904.
[63] Y. Lyu, X. Huang, and Z. Zhang, "EllipsoidNet: Ellipsoid representation for point cloud classification and segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2022, pp. 256–266.
[64] G. Wang, Q. Zhai, and H. Liu, "Cross self-attention network for 3D point cloud," *Knowl.-Based Syst.*, vol. 247, 2022, Art. no. 108769.
[65] C. Zhang, H. Wan, X. Shen, and Z. Wu, "PatchFormer: An efficient point transformer with patch attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 11789–11798.

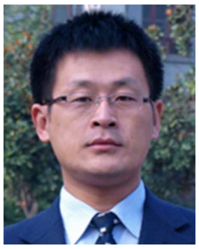
[66] Y. Ben-Shabat, M. Lindenbaum, and A. Fischer, "3DmFV: Three-dimensional point cloud classification in real-time using convolutional neural networks," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3145–3152, Oct. 2018.
[67] Y. Li et al., "PointCNN: Convolution on X-transformed points," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 820–830.
[68] S. Qiu, S. Anwar, and N. Barnes, "Dense-resolution network for point cloud classification and segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2021, pp. 3812–3821.
[69] J. Yang et al., "Modeling point clouds with self-attention and Gumbel subset sampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3318–3327.
[70] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9276–9285.

Yue Wu (Member, IEEE) received the B.Eng. and Ph.D. degrees from Xidian University, Xi'an, China, in 2011 and 2016, respectively. Since 2016, he has been a Teacher with Xidian University. He is currently an Associate Professor with Xidian University. He has authored or coauthored more than 100 papers in refereed journals and proceedings. His research interests include computational intelligence and its applications. He is the Secretary General of Chinese Association for Artificial Intelligence-Youth Branch, Secretary General of CCF Xi'an during 2020–2022, Chair of CCF YOCSEF Xi'an during 2021–2022, CCF/CAAI Senior Member. He is Editorial Board Member for over six journals, including *CAAI Transactions on Intelligence Technology, Frontiers of Computer Science*, *Remote Sensing*, *Electronics*.

Jiaming Liu received the B.S. degree in software engineering from the Jiangxi University of Finance and Economics, Nanchang, China, in 2021. He is currently working toward the master's degree with the School of Computer Science and Technology, Xidian University, Xi'an, China. His research interests include artificial intelligence, machine learning and computer vision.

Maoguo Gong (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees from Xidian University, Xi'an, China, in 2003 and 2009, respectively. Since 2006, he has been a Teacher with Xidian University. He was promoted to an Associate Professor and a Full Professor, in 2008 and 2010, respectively, with exceptive admission. He has authored or coauthored more than 100 articles in journals and conferences. He holds more than 20 granted patents as the first inventor. He is leading or has completed more than twenty projects as the Principle Investigator, funded by the National Natural Science Foundation of China, the National Key Research and De-velopment Program of China, and others. His research interests include computational intelligence, with applications to optimization, learning, data mining, and image understanding. Prof. Gong is the Executive Committee Member of Chinese Association for Artificial Intelligence and a Senior Member of Chinese Computer Federation. He was the recipient of the prestigious National Program for Support of the Leading Innovative Talents from the Central Organization Department of China, the Leading Innovative Talent in the Science and Technology from the Ministry of Science and Technology of China, the Excellent Young Scientist Foundation from the National Natural Science Foundation of China, the New Century Excellent Talent from the Ministry of Education of China, and the National Natural Science Award of China. He is an Associate Editor or an Editorial Board Member for over five journals including IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

Zhixiao Liu received the B.S. degree in communication engineering from Nanchang University, Nanchang, China, in 2021. He is currently working toward the master's degree with the School of Electronic Information, Harbin Engineering University, Harbin, China. His research interests include artificial intelligence, deep learning and computer vision.


Qiguang Miao (Senior Member, IEEE) received the M.Eng. and Doctor degrees in computer science from Xidian University, Xi'an, China. He is currently a Professor with the School of Computer Science and Technology, Xidian University. His research interests include intelligent image processing and multiscale geometric representations for images.

Wenping Ma (Senior Member, IEEE) received the B.S. degree in computer science and technology and the Ph.D. degree in pattern recognition and intelligent systems from Xidian University, Xi'an, China, in 2003 and 2008, respectively. Since 2006, she has been with the Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education, Xidian University, where she is currently an Associate Professor. She has authored or coauthored more than 30 SCI papers in international academic journals, including IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON IMAGE PROCESSING, *Information Sciences, Pattern Recognition, Applied Soft Computing, Knowledge-Based Systems*, *Physica A-Statistical Mechanics and its Applications*, and IEEE GEOSCIENCE AND REMOTE SENSING LETTERS. Her research interests include natural computing and intelligent image processing. Dr. Ma is a member of the Chinese Institute of Electronics and the China Computer Federation.