Dual Transformer for Point Cloud Analysis

Xian-Feng Han, Member, IEEE, Yi-Fei Jin, Hui-Xian Cheng, and Guo-Qiang Xiao

Abstract—Feature representation learning is a key component in 3D point cloud analysis. However, the powerful convolutional neural networks (CNNs) cannot be applied due to the irregular structure of point clouds. Therefore, following the tremendous success of transformer in natural language processing and image understanding tasks, in this paper, we present a novel point cloud representation learning architecture, named Dual Transformer Network (DTNet), which mainly consists of Dual Point Cloud Transformer (DPCT) module. Specifically, by aggregating the well-designed point-wise and channel-wise self-attention models simultaneously, DPCT module can capture much richer contextual dependencies semantically from the perspective of position and channel. With the DPCT model as a fundamental component, we construct the DTNet for performing 3D point cloud analysis in an end-to-end manner. Extensive quantitative and qualitative experiments on publicly available benchmarks demonstrate the effectiveness of our transformer framework for the tasks of 3D point cloud classification, segmentation and visual object affordance understanding, achieving highly competitive performance in comparison with the state-of-the-art approaches.

Index Terms—Classification, point cloud, segmentation, self attention, transformer, visual affordance understanding.

I. INTRODUCTION

ollowing the rapid development of 3D acquisition devices, 3D point clouds play an increasingly important role in a wide range of applications, such as multimedia interaction [1], [2], [3], virtual/augmented reality [4], [5], autonomous driving [6], [7], and robotics [8], [9]. Therefore, how to perform effective point cloud analysis becomes an urgent requirement. Recently, deep learning techniques achieve tremendous success in 2D computer vision domain, which actually provide an opportunity for better understanding of point clouds. However, different from structured 2D images, point clouds are irregular and disordered, making it unreasonable to directly apply the traditional convolutional neural networks to them.

In response to this problem, several state-of-the-art approaches attempt to transform the unstructured point clouds into either voxel grids [10] or multi-views [11], then applying 3D CNNs or 2D CNNs for powerful feature learning. Although promising performances have been reported, these methods suffer from growing memory requirement, high computation cost as well as geometric information loss during transformation. Alternatively, following the great success of pioneering work PointNet [12], the family of point-based approaches runs directly on raw point clouds to learn 3D representation via shared Multi-Layer Perceptrons (MLPs) [13] or well-designed 3D convolutional kernels [14] or graph convolutions [15]. Nevertheless, they fail to capture long-range contextual correlations among points.

More recently, Transformer architecture has achieved a series of breakthroughs in the field of Natural Language Processing (NLP) and 2D Computer Vision [16], [17]. As its core component, self-attention mechanism, on the one hand, is able to learn much richer contextual representations by capturing long-range dependencies in the input sequence. On the other hand, it is invariant to points permutations. Transformer, therefore, is an ideal model suitable for point cloud processing.

Inspired by the superior performance of Transformer, we introduce a well-designed end-to-end architecture, named Dual Transformer Network (DTNet) for 3D point cloud analysis. Its fundamental component is our proposed Dual Point Cloud Transformer (DPCT) structure, which has the ability to aggregate the long-range spatial and channel context-dependencies of point-wise feature maps for feature representation enhancement. Specifically, a DPCT module consists of two parallel branches. One is point-wise multi-head self-attention attempting to spatially extract contextual information among point-wise features, while the other is channel-wise multi-head self-attention model for capturing context dependencies in channel dimension. The outputs of these two attention modules are then concatenated via element-wise sum operation to improve the power of representation. Finally, we construct an U-Net like architecture using multi-scale DPCT modules to perform 3D point cloud analysis and understanding tasks, as shown in Fig. 1.
Quantitative and qualitative evaluations conducted on challenging benchmark datasets demonstrate the effectiveness of our proposed model. And the superior performance further reflects the high competitiveness of our DTNet against other state-of-the-art point cloud learning networks on tasks of 3D shape classification, segmentation and visual object affordance understanding.

In summary, we make the following contributions:

1) A carefully-designed Dual Point Cloud Transformer module based on multi-head self-attention is proposed for 3D point cloud processing. This module can semantically enhance the representation power of learned point features by modeling long-range contextual correlations from the position and channel points of view.

2) Based on our Dual Point Cloud Transformer model, an end-to-end Dual Transformer Network (DTNet) is constructed, which directly operates on 3D point clouds for highly effective feature learning.

3) Extensive evaluations performed on three challenging benchmarks demonstrate the effectiveness of our point cloud Transformer architecture, which achieves competitive performance on 3D object classification, segmentation and 3D visual affordance understanding tasks.

The rest of our paper is structured as follows. Section II presents an overview of recent state-of-the-art works on related topics. Section I introduces the details of our Dual Transformer Network. Quantitative and qualitative experiments as well as analysis are made in Section III-A. Section V draws the conclusions.

II. RELATED WORK

A. Deep Learning on Point Clouds

Motivated by the outstanding performance of deep learning techniques in 2D image understanding tasks, more and more attention has been paid to their applications on 3D point clouds for effective and efficient representation learning. However, the standard operations in CNNs are only suitable for data defined over regular grids [18], which is inappropriate for highly irregular point clouds. In order to address this issue, recent studies based on different input data formats have been proposed, which can be divided into volumetric-based, multi-view based, and point-based methods.

Volumetric-based methods [10], [19] convert the unstructured point clouds into regular volumetric occupancy grids by quantization, which allow the usage of 3D convolutional neural networks (3D CNNs) for feature extraction. However, these approaches suffer from cubic growth of computational complexity and memory requirement with respect to the input resolution, as well as geometric information loss. To overcome these limitations, hierarchical data structure-based methods [6], such as OctNet [20] and Kd-Net [21], have been proposed to concentrate on informative voxels while skipping the empty ones [22]. PointGrid [19] improves the local geometric details learning by incorporating points within each grid.

Multi-view based methods aim to turn 3D problem into 2D problem by projecting the point cloud space into a collection of 2D images with multiple views, so that 2D CNNs can be applied to perform feature learning. The pioneering work MVCNN [23] simply leverages the max-pooling operation to aggregate the multi-view features into a global descriptor. Wei et al. [24] designed a directed graph by treating the views as graph nodes to construct the View-GCN. Although methods of this type have obtained remarkable performance on tasks like object classification [25], [26], [27] with the well-established image CNNs, it is still difficult to determine the appropriate number of views to cover the 3D objects while avoiding information loss and high computational consumption.

Point-based methods directly take the raw point clouds as input for learning 3D representations without any voxelization or projection. As a pioneering work, PointNet [12] learns point-wise features from input point clouds via shared MLPs and obtains a global representation with max-pooling operation that can achieve permutation invariance. PointNet++ [13] extends PointNet by integrating a hierarchical structure that takes both global information and local details into consideration. Subsequently, the works begin to focus on defining explicit convolution kernels for points. KPConv [14] presents a deformable point convolution using a collection of learnable kernel points. FPConv [28] runs directly on local surface of point cloud in a projection-interpolation manner for efficient feature learning. Similar to these methods, our DTNet also deals with 3D point clouds directly without transformations of any intermediate representations.

B. Transformers in Vision

More recently, Transformer networks [29] have made significant progress in Natural Language Processing domain [30], [31] and have achieved astounding performance. Its great success attracts increasing interest among computer vision researchers. They have attempted to transfer these models for 2D image tasks, such as classification [32] and detection [17]. Actually, the core factor of Transformer is self-attention mechanism that has capability of explicitly modeling interactions between elements in the input sequence. iGPT [16] uses transformer for image generation tasks with unsupervised training mechanism. Vision Transformer (ViT) [32] replaces the standard convolutions with transformers to perform large-scale supervised image classification. DETR [17] is the first work to solve the detection issue using transformer model from the perspective of prediction problem. Additionally, since self-attention does not depend on input order, and 3D point clouds can be treated as a set of points with positional attributes, transformer is ideally-suited for point cloud analysis. Therefore, we develop a point cloud transformer network with our carefully-designed Dual Point Cloud Transformer layer, which can correlate the point representations for learning much broader geometry and context dependencies.

III. PROPOSED APPROACH

In this section, we first introduce the general formulation of our proposed Dual Point Cloud Transformer (DPCT) module. This module mainly adopts the essential idea of Transformer models to capture point sets interactions in terms of position
and channel dependencies (shown in Fig. 2). Then, we stack the DPCT modules with increasing receptive fields to construct an end-to-end architecture, termed Dual Transformer Network (DTNet), for point cloud analysis. The overall framework of our DTNet is illustrated in Fig. 3.

### A. Dual Point Cloud Transformer

Given a point cloud with $N$ points $\mathcal{P} = \{p_i \in \mathbb{R}^D, i = 1, 2, \ldots, N\}$, we aim to find a set function $f$ with property of permutation invariance to input, which can project the input $\mathcal{P}$ to a long-range context enhancement feature space $\mathcal{F}$, $f : \mathcal{P} \rightarrow \mathcal{F}$, where $D = 3 + d$ denotes the dimensions of point-wise attributes that may describe the 3D coordinates and additional information (e.g., RGB color, surface normal vector). Actually, Transformer and its associated self-attention mechanism are particularly suitable for this problem due to the usage of matrix multiplication and summation operations. They have the ability to highlight long-term relationships among elements. We, therefore, develop a Dual Point Cloud Transformer block based on the advantage of transformer frameworks, which mainly consists of two individual multi-head self-attention branches for learning interactions across points and channels simultaneously.

**Point-wise Self-attention:** In order to investigate the spatial correlations among points and generate long-range context-dependent representation, we construct a point-wise multi-head self-attention module for enhancement of feature capability. As illustrated in Fig. 2, we formulate our attention model as:

$$F_{p_1}^{l+1} = A_{pa}(F^l) = CN(\mathcal{A}T^1_{l+1}, \mathcal{A}T^2_{l+1}, \ldots, \mathcal{A}T^M_{l+1}) + F^l$$  \hspace{1cm} (1)
Where $\mathcal{F}^l \in R^{N \times C}$ is a point-wise feature map from the $l$th layer. $\mathcal{F}^{l+1}_{pa}$ stands for the output feature map of our point-wise attention model. $A_{pa}(\cdot)$ denotes the point-wise multi-head self-attention operation. $CN(\cdot)$ refers to concatenation operator. $M$ represents the number of self-attention blocks. $\mathcal{A}T^m_{l+1}$ operation is defined as:

$$\mathcal{A}T^m_{l+1}(\mathcal{F}^l) = S^m_{l+1}V^m_{l+1} = \sigma\left(Q^m_{l+1}(K^m_{l+1})^T / \sqrt{C/M}\right)V^m_{l+1}$$  \hspace{1cm} (2)

$$Q^m_{l+1} = \mathcal{F}^lW^m_{Q_{l+1}}$$ \hspace{1cm} (3)

$$K^m_{l+1} = \mathcal{F}^lW^m_{K_{l+1}}$$ \hspace{1cm} (4)

$$V^m_{l+1} = \mathcal{F}^lW^m_{V_{l+1}}$$ \hspace{1cm} (5)

The $m$ is the index of attention heads, $S^m_{l+1} \in R^{N \times N}$ is the point-wise attention matrix of the $m$th head, measuring the points’ impacts on each other. $\sigma$ is the softmax function. $W^m_{Q_{l+1}} \in R^{C \times d_q}$, $W^m_{K_{l+1}} \in R^{C \times d_k}$, $W^m_{V_{l+1}} \in R^{C \times d_v}$ are learnable weight parameters of three linear layers, where we set $d_q = d_k = d_v = C/M$, $C$ is the number of channels.

From (1) and (2), we can reach the conclusion that the output point-wise feature map can be considered as a sum of features assembled from all points and the initial input, which spatially describes the long-range dependencies by integrating contextual information based on attention map.

**Channel-wise Self-attention:** As discussed in PointNet++ [13] and TANet [33], it can be clearly reported that different channels actually contain different geometric/semantic representations, which contribute to the high expressiveness of learned features. Therefore, we attempt to highlight the interactions across different feature channels via the introduction of channel-wise multi-head self-attention model (shown in the bottom branch of Fig. 2). Here, we adopt the similar strategy as that for calculating point-wise attention. Specifically, our channel-wise self-attention model is formally defined as:

$$\mathcal{F}^{l+1}_{ca} = A_{ca}(\mathcal{F}^l)$$

$$\mathcal{A}T^1_{l+1}, \mathcal{A}T^2_{l+1}, \ldots, \mathcal{A}T^M_{l+1} + \mathcal{F}^l$$ \hspace{1cm} (6)

$$\mathcal{A}T^m_{l+1}(\mathcal{F}^l) = U^m_{l+1}v^m_{l+1} = \sigma\left((q^m_{l+1})^T k^m_{l+1} / \sqrt{C/M}\right)v^m_{l+1}$$ \hspace{1cm} (7)

$$q^m_{l+1} = \mathcal{F}^lW^m_{q_{l+1}}$$ \hspace{1cm} (8)

$$k^m_{l+1} = \mathcal{F}^lW^m_{k_{l+1}}$$ \hspace{1cm} (9)

$$v^m_{l+1} = \mathcal{F}^lW^m_{v_{l+1}}$$ \hspace{1cm} (10)

Where $A_{ca}(\cdot)$ indicates our channel-wise multi-head self-attention operation, and its output is $\mathcal{F}^{l+1}_{ca}$. $U^m_{l+1} \in R^{d_c \times d_c}$ denotes the channel attention matrix, indicating the channel’s importance to each other. $W^m_{q_{l+1}} \in R^{C \times d_q}$, $W^m_{k_{l+1}} \in R^{C \times d_k}$, $W^m_{v_{l+1}} \in R^{C \times d_v}$ are weight matrices of fully-connected layers, $d_c = C/M$.

Similar to point-wise attention, we sum the original input and features across all channels to obtain the final channel-wise attention feature map. And this map indeed encodes much wider range of channel-wise relationships via the application of self-attention mechanism, which gives an improvement of contextual representation.

Finally, in order to aggregate the long-range spatial and channel contextual information, we perform the element-wise addition between the resulting point-wise and channel-wise attention features. Specifically, the fused feature $\mathcal{F}^{l+1}_{DPCT}$ can be formulated as,

$$\mathcal{F}^{l+1}_{DPCT} = \mathcal{F}^{l+1}_{pa} \oplus \mathcal{F}^{l+1}_{ca}$$ \hspace{1cm} (11)

Where $\oplus$ implies the element-wise summation. As shown in the left part of Fig. 2, we build the Dual Point Cloud Transformer (DPCT) model based on our well-designed attention modules, which can strengthen the discriminability of learned contextual representations.

**B. Architecture of Dual Transformer Network**

Based on our proposed Dual Point Cloud Transformer (DPCT) block, we construct the complete deep networks for 3D point cloud analysis including classification, segmentation and visual affordance prediction, where segmentation and affordance understanding share the same architecture but with different classifier modules. Specifically, from Fig. 3, it can be obviously seen that the networks directly take 3D point clouds as input, then progressively perform feature learning with stacked Point Feature Down Sample (FDS) layer, Dual Point Cloud Transformer (DPCT) layer, Point Feature Up Sample (FUS) Layer and Fully-connected(FC) layer, where our DPCT is the critical component for feature aggregation in our networks. **Point Feature Down Sample Layer:** In order to construct a hierarchical feature for multi-scale analysis, a Point Feature Down Sample (FDS) layer is added before our DPCT, which is used to downsample the point-wise feature maps as required. Specifically, given an input feature map $\mathcal{F}^l$ from the $l$th layer, the farthest point sampling algorithm (FPS) is performed to generate another feature map $\mathcal{F}^l$, a subset of $\mathcal{F}^l$, then we group $K$ neighboring points in a r-ball local region for each centroid point of $\mathcal{F}^l$, following by a linear transformation, batch normalization (BN) and ReLU operations. This FDS layer can be formally defined as:

$$\mathcal{F}^{l+1} = \text{Relu}(BN(W^l(\text{Agg}(FPS(\mathcal{F}^l))))))$$ \hspace{1cm} (12)

Where $\text{Agg}(\cdot)$ indicates the multi-scale feature aggregation operation. $W^l$ denotes the learnable weight parameters of linear transformation.

**Point Feature Up Sample Layer:** For segmentation and visual affordance recognition tasks, we should make dense prediction. Therefore, we choose to put well-designed Point Feature Up Sample (FUS) Layers in the decoding stage, which can help to progressively increase the resolution of point-wise feature map to the original input size. Here, K nearest neighbors interpolation is adopted to upsample point set according to Euclidean distance. Similar to PointNet++ [13], we resort to skip connection and linear transformation together with batch normalization and ReLU for feature fusion between encoder and decoder. The
Our Transformer network for 3D object classification. Here, we make full use of the U-Net structure as backbone. Specifically, the number of layers in encoder and decoder, hyperparameters like downsampling rates are set differently for each specific task and will be given detailedly in Experiment Section.

IV. EXPERIMENTS

In this section, we evaluate the effectiveness and performance of our DTNet on three publicly available datasets, namely ModelNet [34], ShapeNet [35] and 3D AffordanceNet [36], for tasks of classification, segmentation, and 3D affordance estimation, respectively. In addition, our point cloud transformer models are implemented with PyTorch framework on a single NVIDIA TITAN RTX 24 G GPU. The weight decay is set to 0.0001.

A. Point Cloud Classification

Dataset and Metric: We evaluate our 3D object classification transformer model on ModelNet40 [34], which consists of 12,311 CAD models from 40 classes with 9,843 shapes for training and 2,468 objects for testing. Following PointNet [12], 1,024 points are uniformly sampled from each model. During training, we perform data augmentation by adopting a random point dropout, a random scaling in [-0.66, 1.5] and a random shift in [-0.2, 0.2]. The overall accuracy (OA) is treated as evaluation metric for classification task.

Network Configuration: Our Transformer network for 3D shape classification is shown in the bottom of Fig. 3. We use three Point Feature Down Sample layers, each associating with an individual Dual Point Cloud Transformer module. Three fully-connected layers are appended at the end of our model. The network configuration details are summarized as follows: INPUT(N=1,024, C=3) → FDS(N₀=512, C₁=320) → DPCT(N₀=128, C₁=320, C₂=640) → DPCT(N₀=128, C₁=640, C₂=640) → FDS(N₀=1, C₁=643, C₂=1,024) → DPCT(N₀=1, C₁=1,024, C₂=1,024) → FC(C₁=1,024, C₂=512) → FC(C₁=512, C₂=256) → FC(C₁=256, C₂=40), where N₀ denotes the number of output points. Each layer takes Cᵢ input features and generates Cᵢ features. The network is trained for 150 epochs with a batch size of 32. The initial learning rate is set to 0.01 and is dynamically adjusted to 0.0001 using cosine annealing strategy. For optimization, we choose to use Stochastic Gradient Decent (SGD) with a momentum of 0.9 as the optimizer for training.

Performance Comparison: To verify the effectiveness of our DTNet, we perform a comparison with several representative state-of-the-art models, and quantitative performance results are reported in Table I. From these results, it can be clearly seen that (1) our DTNet achieves a competitive overall accuracy of 93.4%, outperforming almost all point-based methods taking only 1,024 points as input. (2) Particularly, compared with PointNet++ and its dense (5 k points) version, we make an improvement of 2.7% and 1.5% in terms of OA, respectively. (3) Although SO-Net also obtains a superior result (93.4%), it uses 5 k points with additional normal information as input while our DTNet taking only 1,024 points.

B. Point Cloud Part Segmentation

Dataset and Metric: The object part segmentation task can be treated as a per-point classification problem. We choose to train and test our segmentation DTNet on ShapeNet Part benchmark dataset [35], which contains 16,881 objects from 16 different classes with a total of 50 part-level labels. For experimental studies, we follow the officially defined 14,007/2,874 training/testing split. And 2,048 points are sampled from each shape as input. In addition, we perform the same data augmentation as that for classification. The standard evaluation metrics, including mean IoU over all part classes and category-wise IoU, are computed to report the performance.

Network Configuration: The network for part segmentation is presented in the top of Fig. 3. It is a U-Net like architecture. For encoder, we adopt the same structure as classification network, while in the decoding stage, three feature propagation blocks (combination of Point Feature Up Sample layer and DPCT layer) are appended. The detailed configurations for each layer are set

| Method                  | Representation | Input Size | ModelNet40 |
|-------------------------|----------------|------------|-------------|
| DeepPoint [37]          | Points         | 5000 × 3   | 90.0%       |
| ECC [38]                | Points         | 1000 × 3   | 83.2%       |
| DGCNN [39]              | Points         | 1024 × 3   | 92.2%       |
| Kd-Net [21]             | Points         | 215 × 3    | 85.8%       |
| KPConv [14]             | Point          | 7000 × 3   | 92.9%       |
| PointNet [12]           | Points         | 1024 × 3   | 89.2%       |
| PointNet++ [13]         | Points         | 1024 × 3   | 90.7%       |
| 3DmpFC-Net [40]         | Points         | 2048 × 3   | 91.4%       |
| FoldingNet [41]         | Points         | 2048 × 3   | 88.4%       |
| KC-Net [42]             | Points         | 1024 × 3   | 91.0%       |
| PointCNN [43]           | Points         | 1024 × 3   | 92.5%       |
| PCNN [44]               | Points         | 1024 × 3   | 92.3%       |
| RGCNN [55]              | Points         | 1024 × 3   | 90.5%       |
| ShapeContextNet [46]    | Points         | 1024 × 3   | 90.0%       |
| Spec-GCN [47]           | Points         | 2048 × 3   | 92.1%       |
| SRN [48]                | Points         | 1024 × 3   | 91.5%       |
| Point2Node [49]         | Points         | 1000 × 3   | 93.0%       |
| Point2Sequence [50]     | Points         | 1024 × 3   | 92.6%       |
| PointConv [51]          | Points         | 1024 × 3   | 92.5%       |
| Ψ-CNN [52]              | Points         | -          | 92.0%       |
| FPCConv [28]            | -              | -          | 92.5%       |
| Point Transformer [53]  | Points         | 1024 × 3   | 92.6%       |
| SPHID-GCN [18]          | Points         | 1000 × 3   | 92.1%       |
| DR-Net [54]             | Points         | 1024 × 3   | 93.1%       |
| Point Transformer [55]  | Points         | 1024 × 3   | 92.8%       |
| PT [53]                 | Points         | 1024 × 3   | 92.9%       |
| PCT [56]                | Points         | 1024 × 3   | 93.2%       |
| PRA-Net [57]            | Points         | 1024 × 3   | 93.2%       |
| PACConv [38]            | Points         | 1024 × 3   | 93.2%       |
| DTNet                   | Points         | 1024 × 3   | 93.4%       |
| PointNet++ [13]         | Points+normals | 5000 × 6   | 91.9%       |
| SO-Net [59]             | Points+normals | 5000 × 6   | 93.4%       |
| SpiderCNN [60]          | Points+normals | 1024 × 6   | 92.4%       |
| SPCNN [61]              | Points+normals | 1024 × 6   | 92.3%       |
| PointWeb [62]           | Points+normals | 1024 × 6   | 92.3%       |
| ELM [63]                | Points+normals | 2048 × 6   | 93.2%       |
| FPCConv [28]            | Points+normals | -          | 92.5%       |
For object affordance estimation on...CrossMI5643...as: INPUT(NHAN et al.: DUAL TRANSFORMER FOR POINT CLOUD ANALYSIS

| Method                   | mIoU | aeros. | bags. | caps. | cars. | chairs. | ep. | guitars. | knives. | lamps. | laptops. | motors. | mugs. | pistols. | rockets. | state | table |
|--------------------------|------|--------|-------|-------|-------|---------|-----|----------|---------|--------|----------|---------|-------|----------|----------|-------|-------|
| ShapeNet [35]            | 81.4 | 81.0   | 78.4  | 77.7  | 75.7  | 87.6    | 61.9| 92.0     | 85.4    | 82.3   | 93.7     | 70.6    | 91.9 | 65.9     | 53.1     | 69.8  | 75.3  |
| PointNet [12]            | 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++ [13]          | 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  |
| KD-Net [21]              | 82.3 | 80.1   | 74.6  | 74.3  | 70.3  | 86.5    | 73.5| 90.2     | 87.2    | 71.0   | 94.9     | 57.4    | 86.7 | 71.8     | 51.8     | 69.9  | 80.3  |
| SO-Net [59]              | 84.9 | 82.8   | 77.8  | 88.0  | 77.3  | 90.6    | 73.5| 90.7     | 83.9    | 82.8   | 94.8     | 69.1    | 94.2 | 80.9     | 53.1     | 72.9  | 83.0  |
| RGCNN [45]               | 84.3 | 80.2   | 82.6  | 92.6  | 75.3  | 89.2    | 73.7| 91.3     | 88.4    | 83.3   | 96.0     | 63.9    | 95.7 | 60.9     | 44.6     | 72.9  | 80.4  |
| DGCNN [39]               | 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     | 63.8     | 74.5  | 82.6  |
| SRN [48]                 | 85.3 | 82.4   | 79.8  | 88.1  | 77.9  | 90.7    | 69.6| 90.9     | 86.3    | 84.0   | 95.4     | 72.2    | 94.9 | 81.3     | 62.1     | 75.9  | 83.2  |
| SPCNN [61]               | 85.4 | 83.0   | 83.4  | 87.0  | 80.2  | 91.5    | 79.9| 91.1     | 86.2    | 84.2   | 96.7     | 69.3    | 94.8 | 82.5     | 59.9     | 73.1  | 82.9  |
| PointConv [51]           | 85.7 | 83.9   | 83.7  | 87.1  | 80.3  | 91.5    | 79.5| 91.1     | 86.5    | 84.2   | 96.7     | 69.3    | 94.8 | 82.5     | 59.9     | 73.1  | 82.9  |

Effect of Each Component: To further study the influence of our proposed transformer component, we carry out additional experiments on the classification task of ModelNet40. We remove the DPCT modules from DTNet as the baseline. Table IV presents the results of different design choices of DTNet in terms of accuracy average class (ACC) and overall accuracy (OA). Comparing with the baseline, we can clearly observe that point-wise (PWSA) and channel-wise self-attention (CWSA) make remarkable contributions to point cloud representation learning, achieving the results of 91.0% and 90.6% in OA, respectively. And our DPCT module with integration of these two attention models obtain a significant improvement over baseline methods in 9, 10, 7 and 15 out of 18 categories. Additionally, for affordance categories, such as Open, Sit, Support, Pull and table. We visualize several 3D part segmentation examples in Fig. 4. From these results, we can verify the success of our DTNet for part segmentation task.

C. 3D Visual Affordance Understanding

Dataset and metric: 3D Object Affordance Understanding is an essential task for agent (e.g., robot) to interact with the realistic and complex environment. Its goal is to estimate point-wise affordance contained in each individual point cloud object. 3D AffordanceNet is a newly-proposed and challenging benchmark task.

Effect of Each Component: To further study the influence of our proposed transformer component, we carry out additional experiments on the classification task of ModelNet40. We remove the DPCT modules from DTNet as the baseline. Table IV presents the results of different design choices of DTNet in terms of accuracy average class (ACC) and overall accuracy (OA). Comparing with the baseline, we can clearly observe that point-wise (PWSA) and channel-wise self-attention (CWSA) make remarkable contributions to point cloud representation learning, achieving the results of 91.0% and 90.6% in OA, respectively. And our DPCT module with integration of these two attention models obtain a significant improvement over baseline methods in 9, 10, 7 and 15 out of 18 categories. Additionally, for affordance categories, such as Open, Sit, Support, Pull and table. We visualize several 3D part segmentation examples in Fig. 4. From these results, we can verify the success of our DTNet for part segmentation task.

D. Ablation Study and Analysis

Effect of Each Component: To further study the influence of our proposed transformer component, we carry out additional experiments on the classification task of ModelNet40. We remove the DPCT modules from DTNet as the baseline. Table IV presents the results of different design choices of DTNet in terms of accuracy average class (ACC) and overall accuracy (OA). Comparing with the baseline, we can clearly observe that point-wise (PWSA) and channel-wise self-attention (CWSA) make remarkable contributions to point cloud representation learning, achieving the results of 91.0% and 90.6% in OA, respectively. And our DPCT module with integration of these two attention models obtain a significant improvement over baseline methods in 9, 10, 7 and 15 out of 18 categories. Additionally, for affordance categories, such as Open, Sit, Support, Pull and table. We visualize several 3D part segmentation examples in Fig. 4. From these results, we can verify the success of our DTNet for part segmentation task.
Fig. 4. Qualitative comparisons between our DTNet and the ground truth on ShapeNet Part dataset.
TABLE III
QUANTITATIVE PERFORMANCE COMPARISON WITH STATE-OF-THE-ART MODELS ON 3D AFFORDANCE-NET BENCHMARK. ‘P’, ‘D’, AND ‘U’ STAND FOR POINTNET++, [13], DGCNN [39] AND U-NET [66], RESPECTIVELY. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD.

| Full-Input | Avg | Grape | Left | Open | Rev | Elev | Sift | Support | Warg | Error | Display | Patch | Eta | Wear | Press | Move | Press | Cut | Rate |
|-----------|-----|-------|------|------|-----|------|------|---------|------|-------|---------|------|-----|------|-------|------|-----|------|-----|------|-----|------|
| P-Net     | 54.7| 51.4 | 71.3 | 38.9 | 46.8| 63.6 | 61.5 | 32.3    | 18.0 | 46.7 | 39.7   | 20.5 | 37.9 | 41.6 | 20.4 | 31.4 | 33.3 | 41.4 | 39.7 |
| P-Net++   | 57.4| 52.2 | 72.7 | 41.3 | 49.6| 66.0 | 63.4 | 34.3    | 20.9 | 50.8 | 41.5   | 21.7 | 38.7 | 42.2 | 21.1 | 32.1 | 34.3 | 42.1 | 40.8 |
| D-Net     | 46.6| 43.9 | 63.2 | 51.4 | 51.8| 62.3 | 60.9 | 34.0    | 22.7 | 47.7 | 65.8   | 20.5 | 40.3 | 38.0 | 18.3 | 34.2 | 33.5 | 40.2 | 39.1 |
| U-Net     | 58.5| 55.3 | 86.7 | 83.9 | 81.6| 90.1 | 91.4 | 62.4    | 79.2 | 64.1 | 86.1   | 61.9 | 81.5 | 81.5 | 61.9 | 81.5 | 81.5 | 81.5 | 81.5 |
| D-Net++   | 57.8| 54.3 | 75.2 | 56.4 | 58.8| 68.4 | 60.8 | 35.7    | 19.1 | 45.0 | 61.3   | 18.5 | 56.2 | 57.3 | 20.4 | 55.9 | 54.2 | 54.2 | 54.2 |
| DGCNN     | 56.1| 52.9 | 84.3 | 81.8 | 81.8| 84.2 | 95.8 | 59.7    | 72.9 | 67.1 | 81.4   | 84.0 | 82.9 | 70.3 | 91.6 | 79.2 | 90.8 | 90.8 |
| U-NET++   | 15.7| 13.7 | 41.2 | 22.6 | 40.8| 29.4 | 37.3 | 18.6    | 4.7  | 18.3 | 32.4   | 4.2  | 13.0 | 13.1 | 4.4  | 14.5 | 9.2  | 14.2 | 41.5 |

TABLE IV
ABLATION STUDIES ON THE MODELNET40 DATASET. WE ANALYZE THE EFFECTS OF POINT-WISE SELF-ATTENTION (PWSA) AND CHANNEL-WISE SELF-ATTENTION (CWSA).

| Method          | ACC  | OA   | Time (ms) |
|-----------------|------|------|-----------|
| Baseline        | 87.5 | 90.0 | 29.3      |
| Baseline + PWSA | 88.1 | 91.0 | 29.9      |
| Baseline + CWSA | 88.0 | 90.6 | 29.8      |
| DTNet           | 90.9 | 93.4 | 31.7      |

TABLE V
THE CLASSIFICATION RESULTS ON MODELNET40 USING DIFFERENT UPSAMPLING STRATEGIES.

| Upsampling | ACC  | OA   |
|------------|------|------|
| EVA        | 90.3 | 92.8 | 90.9 | 93.4 |

TABLE VI
COMPLEXITY OF REPRESENTATIVE NETWORKS ON MODELNET40 CLASSIFICATION BENCHMARK. #PARAMS: THE NUMBER OF NETWORK PARAMETERS. #FLOPS: FLOATING POINT OPERATIONS PER SAMPLE.

| Method         | #Params | #FLOPs | OA   |
|----------------|---------|--------|------|
| PointNet [12]  | 3.5M    | 0.44G  | 89.2 |
| PointNet++ [13]| 1.8M    | 4.05G  | 90.7 |
| DGCNN [39]     | 1.8M    | 2.43G  | 92.2 |
| PCT [56]       | 2.9M    | 2.32G  | 93.2 |
| DTNet          | 6.4M    | 4.36G  | 93.4 |

by 3.4%. We also list the average inference time running on testing dataset with a single NVIDIA TITAN RTX 24 G GPU. As it presents, the introduction of PWSA and CWSA models actually makes relatively slight impacts on inference efficiency. These results convincingly validate the effectiveness and benefit of our DPCT block.

Complexity Analysis: We summarize the number of parameters and floating point operations per sample required by various networks in point cloud classification. As reported in Table VI, our DTNet achieves the highest classification performance with a little higher computational cost (#FLOPs) and space efficiency (#Params). Therefore, in our future work, we will consider how to optimize our transformer network to reduce the complexity.

Different Upsampling Strategies: To investigate the impact of different upsampling strategies on 3D object classification task. We choose two upsampling methods, K nearest neighbors interpolation (KNN) [13] and Edge-vector based approximation (EVA) upsampling [67] to study. In this experiment, we use PyTorch to implement the EVA according to its core idea that encodes locally geometric information. We replace the KNN based method in our transformer architecture with EVA, and use the same configurations to train the network. As shown in Table V, KNN based method brings much better performance with 0.6% ACC and OA improvements over EVA.

E. Robustness Analysis

In order to fairly evaluate the robustness of our DTNet with respect to noise, incompleteness and sparsity, we use the same trained model, and test data for all experiments.

Gaussian Noise: We add noise sampled from \( N(0, \sigma^2) \) to test data, which is used to verify the robustness of our model against varying levels of Gaussian noise. Experimental results are illustrated in Fig. 5. Our network acquires relatively satisfactory performance when standard deviation is less than 0.07. These results indicate that DTNet is robust to some extent of Gaussian noise.

Incomplete Input: We change the ratios of missing points to generate inputs with different levels of incompleteness during test time. Fig. 6 demonstrates the ACC and OA performance of DTNet under varying ratios. It can be observed that our DTNet presents a stable performance with the increase of missing data ratio, which implies its robustness to incomplete inputs.

Sparsity: We test proposed DTNet on robustness against density variation. Farthest Sampling algorithm is adopted to obtain
inputs with different sparse sampling densities. As can be witnessed from ACC and OA results from Fig. 7, when we decrease the input point number, DTNet can still perform 3D object classification accurately.

V. CONCLUSION AND DISCUSSION

The recent advent of Transformer provides an solution to point permutation challenge faced by deep learning on point clouds. This paper introduced an end-to-end point cloud analysis transformer model, termed Dual Transformer Network. Its core component is our well-designed Dual Point Cloud Transformer, which can capture long-range context dependencies by investigating the point-wise and channel-wise relationships. Extensive experiments on challenging benchmark datasets show the remarkable effectiveness of our DTNet, achieving state-of-the-art performance on tasks of classification, segmentation and visual affordance understanding.

However, some issues are expected to be explored. At first, data imbalance problem limits the performance of our DTNet for the tasks of segmentation and affordance estimation. For example, from experimental results of segmentation, it can be concluded that DTNet achieves the best performance on categories with large amount of training data (e.g., lamp and table), since the transformer architectures generally rely on large-scale datasets for complex models learning. Secondly, the local feature representation that our DTNet ignores should be taken into account for descriptiveness improvement, which may contribute to much more accurate segmentation and affordance analysis. Finally, although DTNet attains comparable results, it requires higher time and space consumption, which has some negative effects on our network’s performance. Therefore, in the future, we will attempt to solve these problems to boost the performance of 3D point cloud analysis.

REFERENCES

[1] H. Liu, Y. Guo, Y. Ma, Y. Lei, and G. Wen, “Semantic context encoding for accurate 3D point cloud segmentation,” IEEE Trans. Multimedia, vol. 23, pp. 2045–2055, Jul. 2021.
[2] C. Chen, S. Qian, Q. Fang, and C. Xu, “HAPGN: Hierarchical attentive pooling graph network for point cloud segmentation,” IEEE Trans. Multimedia, vol. 23, pp. 2335–2346, 2021.
[3] S. Qiu, S. Anwar, and N. Barnes, “Geometric back-projection network for point cloud classification,” IEEE Trans. Multimedia, vol. 24, pp. 1943–1955, 2021.
[4] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal, “Learning multiview 3D point cloud registration,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1759–1769.
[5] H. Jiang, F. Yan, J. Cai, J. Zheng, and J. Xiao, “End-to-end 3D point cloud instance segmentation without detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 12796–12805.
[6] 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.
[7] 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. 11108–11117.
[8] E. Nezhadarya, E. Taghavi, R. Razani, B. Liu, and J. Luo, “Adaptive hierarchical down-sampling for point cloud classification,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 12956–12964.
[9] Y. Wang and J. M. Solomon, “Deep-closest point: Learning representations for point cloud classification,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 3523–3532.
[10] 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.
[11] 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.
[12] 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. 652–660.
[13] 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. 31st Int. Conf. Neural Inf. Process. Syst., 2017, pp. 5105–5114.
[14] H. Thomas et al., “KPConv: Flexible and deformable convolution for point clouds,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2019, pp. 6411–6420.
[15] Z.-H. Lin, S.-Y. Huang, and Y.-C. F. Wang, “Convolution in the cloud: Learning deformable kernels in 3D graph convolution networks for point cloud analysis,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1800–1809.
[16] M. Chen et al., “Generative pretraining from pixels,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 1691–1703.
[17] N. Carion et al., “End-to-end object detection with transformers,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 213–229.
[18] H. Lei, N. Akhtar, and A. Mian, “Spherical kernel for efficient graph convolution on 3D point clouds,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 10, pp. 3664–3680, Oct. 2021.
[19] T. Le and Y. Duan, “PointGrid: A deep network for 3D shape understanding,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 9204–9214.
[20] G. Riegler, A. O. Usuya, and A. Geiger, “OctNet: Learning deep 3D representations at high resolutions,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 3577–3586.
[21] R. Klokov and V. Lempitsky, “Escape from cells: Deep kd-networks for the recognition of 3D point cloud models,” in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 863–872.
HAN et al.: DUAL TRANSFORMER FOR POINT CLOUD ANALYSIS

[22] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, “Graph attention convolution for point cloud semantic segmentation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 10296–10305.

[23] W. Yin and H. Schütze, “Multichannel variable-size convolution for sentence classification,” in Proc. 19th Conf. Comput. Natural Lang. Learn., 2015, pp. 204–214.

[24] X. Wei, R. Yu, and J. Sun, “View-VCNN: View-based graph convolutional network for 3D shape analysis,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 1850–1859.

[25] C. R. Qi et al., “Volumetric and multi-view CNNs for object classification on 3D data,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 5648–5656.

[26] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao, “GVCNN: Group-view convolutional neural networks for 3D shape recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 264–272.

[27] C. Ma, Y. Guo, J. Yang, and W. An, “Learning multi-view representation with LSTMs for 3-D shape recognition and retrieval,” IEEE Trans. Multimedia, vol. 21, no. 5, pp. 1169–1182, May 2019.

[28] Y. Lin et al., “FPConv: Learning local flattening for point convolution,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 4293–4302.

[29] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial transformer networks,” in Proc. 28th Int. Conf. Neural Inf. Process. Syst., 2015, pp. 2017–2025.

[30] N. Parmar et al., “Image transformer,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 4055–4064.

[31] H. Hu, Z. Zhang, Z. Xie, and S. Lin, “Local relation networks for image recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 3464–3473.

[32] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” in Proc. Int. Conf. Learn. Representations, 2021.

[33] Z. Liu et al., “TA-Net: Robust 3D object detection from point clouds with triple attention,” in Proc. AAAI Conf. Artif. Intell., 2020, pp. 11677–11684.

[34] Z. Wiciak et al., “3D ShapeNets: A deep representation for volumetric shapes,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 1912–1920.

[35] L. Yi et al., “A scalable active framework for region annotation in 3D shape collections,” ACM Trans. Graph., vol. 35, no. 6, 2016, Art. no. 210.

[36] S. Deng, X. Xu, C. Wu, K. Chen, and K. Jia, “3D AffordanceNet: A benchmark for visual object affordance understanding,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2021, pp. 1778–1787.

[37] S. Ravanbakhsh, J. Schneider, and B. Pocock, “Deep learning with sets and point clouds,” in Proc. Int. Conf. Learn. Representations, 2017.

[38] M. Simonovsky and N. Komodakis, “Dynamic edge-conditioned filters in convolutional neural networks on graphs,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 3693–3702.

[39] Y. Wang et al., “Dynamic graph CNN for learning on point clouds,” ACM Trans. Graph., vol. 28, 2019, Art. no. 146.

[40] Y. Ben-Shahar, 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.

[41] Y. Yang, C. Feng, Y. Shen, and D. Tian, “FoldingNet: Point cloud auto-encoder via deep grid deformation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 206–215.

[42] Y. Shen, C. Feng, Y. Yang, and D. Tian, “Mining point cloud local structures by kernel correlation and graph pooling,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 4548–4557.

[43] Y. Li et al., “PointCNN: Convolution on χ-transformed points,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2018, pp. 828–838.

[44] N. Parmar et al., “Image transformer,” in Proc. Int. Conf. Learn. Representations, 2019, pp. 116208–116217.

[45] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, “SpiderCNN: Deep learning on point sets with parameterized convolutional filters,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 87–102.

[46] Y. Yao et al., “Spherical fractal convolutional neural networks for point cloud recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 9397–9406.

[47] X. Li, B. M. Chen, and G. H. Lee, “So-net: Self-organizing network for point cloud analysis,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 9397–9406.

[48] X. Lu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, “SpiderCNN: Deep learning on point sets with parameterized convolutional filters,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 87–102.

[49] Y. Yao, J. Lu, and J. Zhou, “Spherical fractal convolutional neural networks for point cloud recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 452–460.

[50] 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. 5565–5573.

[51] K. Fujiwara and T. Hashimoto, “Neural implicit embedding for point cloud classification,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 11734–11743.

[52] X. Xu and G. H. Lee, “Weakly supervised semantic point cloud segmentation: Towards 10x fewer labels,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 13706–13715.

[53] X. Yan, C. Zhang, Z. Li, S. Wang, and S. Cui, “PointASNet: Robust point clouds processing using nonlocal neural networks with adaptive sampling,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 5588–5597.

[54] S. Xie et al., “PointContrast: Unsupervised pre-training for 3D point cloud understanding,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 574–591.

[55] L. Luo, L. Tan, W. Zhou, S. Wang, and Z.-X. Yang, “PU-EVA: An edge-vector based approximation solution for flexible-scale point cloud upsampling,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., 2021, pp. 16208–16217.
Yi-Fei Jin is currently working towards the B.E. degree from the College of Computer and Information Science, Southwest University, Chongqing, China. Her research interests include 3D point cloud processing.

Hui-Xian Cheng received the B.S. degree from the Wuhan University Of Technology, Wuhan, China, in 2019. He is currently working the M.S. degree from the College of Computer and Information Science, Southwest University, Chongqing, China. His research interests include 3D scene understanding, point cloud semantic segmentation, and multi-modal fusion.

Guo-Qiang Xiao received the Ph.D. degree in signal and information processing from the University of Electronic Science and Technology of China, Chengdu, China, and the B.E. degree in radio technology from Chongqing University, Chongqing, China, in 1999 and 1986, respectively. Since 1986, he has been with the College of Computer and Information Science, Southwest University, Chongqing, China, where he is currently a Professor. His research interests include computer vision, pattern recognition, machine learning, and Big Data mining.