FG-Net: A Fast and Accurate Framework for Large-Scale LiDAR Point Cloud Understanding

Kangcheng Liu, *Student Member, IEEE*, Zhi Gao, Feng Lin, and Ben M. Chen, *Fellow, IEEE*

***Abstract*—This work presents FG-Net, a general deep learning framework for large-scale point cloud understanding without voxelizations, which achieves accurate and real-time performance with a single NVIDIA GTX 1080 8G GPU and an i7 CPU. First, a novel noise and outlier filtering method is designed to facilitate the subsequent high-level understanding tasks. For effective understanding purpose, we propose a novel plug-and-play module consisting of correlated feature mining and deformable convolution-based geometric-aware modeling, in which the local feature relationships and point cloud geometric structures can be fully extracted and exploited. For the efficiency issue, we put forward a new composite inverse density sampling (IDS)-based and learning-based operation and a feature pyramid-based residual learning strategy to save the computational cost and memory consumption, respectively. Compared with current methods which are only validated on limited datasets, we have done extensive experiments on eight real-world challenging benchmarks, which demonstrates that our approaches outperform state-of-the-art (SOTA) approaches in terms of accuracy, speed, and memory efficiency. Moreover, weakly supervised transfer learning is also conducted to demonstrate the generalization capacity of our method.**

***Index Terms*—3-D scene classification, 3-D semantic segmentation, large-scale point cloud understanding, scene understanding in robotics, weakly supervised transfer learning.**

I. Introduction

DUE TO the directness and robustness in obtaining 3-D information, there has been an increasing proliferation of light detection and ranging (LiDAR) sensors which have been widely deployed on a variety of intelligent agents, such as unmanned ground vehicles (UGVs), unmanned aerial vehicles (UAVs) to perform localization, obstacle detection, exploration, etc. Consequently, efficient and effective large-scale 3-D LiDAR point cloud understanding is of great importance to facilitate machine perception, which bridges the gap between 3-D points and any high-level information, structural or semantic, or both [1]–[3]. However, due to the electrical and mechanical disturbances, and the reflectance property of targets, the point clouds often suffer from noise and outliers. Moreover, compared with the 2-D raster image, the topological relationship between objects in 3-D point clouds is much weakened, rendering the task of segmentation and understanding much more challenging. Therefore, autonomous large-scale point cloud understanding remains an open problem and requires urgent efforts to tackle the challenges, especially when both accuracy and efficiency are taken into account.

Like any high-level task in 2-D image domain, such as object detection, segmentation, or classification, the point cloud understanding methods can also be classified into traditional category and deep learning-based methods. In the traditional category, the representative histogram-based methods [6], [7] encode the *k*-nearest-neighbor (kNN) geometric features of a 3-D point via calculating its surrounding multidimensional average curvature for local geometric variations descriptions. In signature-based [8] and transform-based [2] methods, handcrafted feature descriptions of point clouds have been proposed and exploited for semantic understanding. However, the performances of these methods are merely demonstrated in well-controlled conditions with ideal assumptions, such as noise-free and homogeneous environments [2]. On the other hand, deep learning-based point cloud processing methods have been proposed with promising results in recent years. The mainstream point cloud understanding methods can be roughly divided into three categories: 1) projection based; 2) voxel based; and 3) direct point based. Representative works of each category will be discussed in Section II, while limitations of these methods are summarized here: first, the sampling operation of all these methods has high computational cost and memory consumption. For instance, the widely applied farthest point sampling (FPS) [9], [10] takes more than 1000 s to subsample 10^5 points to about 10^3 points. Furthermore, their subsequent perception networks usually rely on the expensive operations, such as voxelizations [11], [12] or graph construction [13]. Second, nearly all existing methods are designed for small-scale point clouds without considering noise and outliers which are inevitable in practice. Moreover, the large-scale point clouds typically

Manuscript received 13 July 2021; revised 3 January 2022; accepted 11 March 2022. Date of publication 13 April 2022; date of current version 21 December 2022. This work was supported in part by the Hong Kong Centre for Logistics Robotics; in part by the National Natural Science Foundation of China under Grant 42192583; in part by the Hubei Province Natural Science Foundation under Grant 2021CFA088; and in part by the Wuhan University-Huawei Geoinformatics Innovation Laboratory & CAAI-Huawei MindSpore Open Fund. This article was recommended by Associate Editor D. Zhao. *(Corresponding author: Kangcheng Liu.)*

Kangcheng Liu and Ben M. Chen are with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: kcliu@mae.cuhk.edu.hk; bmchen@cuhk.edu.hk).

Zhi Gao is with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430070, Hubei, China (e-mail: gaozhinus@gmail.com).

Feng Lin is with the Frontier Academic Center, Peng Cheng Laboratory, Shenzhen 518066, Guangdong, China (e-mail: truelinfeng@outlook.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2022.3159815.

Digital Object Identifier 10.1109/TCYB.2022.3159815

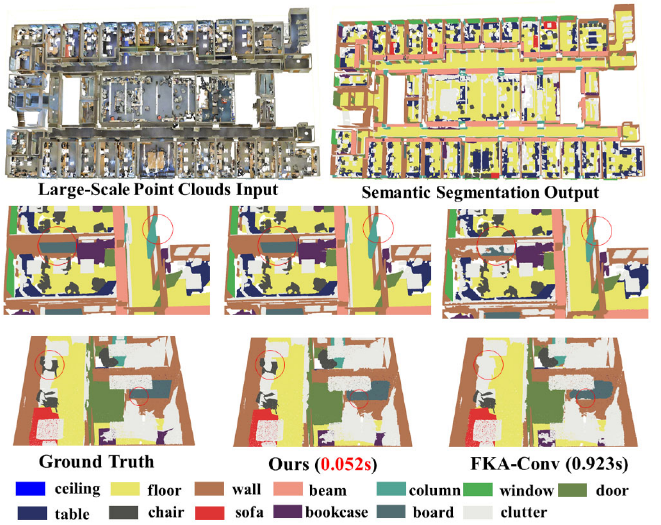


Fig. 1. Semantic segmentation results of our method compared with the SOTA method FKA-Conv [4] on S3DIS [5]. The top row shows the overall segmentation performance by our method. The bottom two rows show the detailed comparisons of segmentation performance highlighted by red circles. Our method achieves real-time segmentation performance of 0.052 s per 10^5 points, which is better and faster than the SOTA method FKA-Conv.

suffer from great class imbalance in semantic categories, and points obtained by LiDAR in complex dynamic environments are often irregular, orderless, and have distant distributed semantic information. For example, in autonomous driving scenarios, typical objects exhibit diverse geometric shapes with varying object sizes (e.g., cyclists and persons) or have distribution across a long spatial range in a nonuniform way (e.g., road, buildings, and vegetations). However, to the best of our knowledge, the existing methods can hardly capture complex geometries or latent feature correlations in large-scale point clouds effectively.

To overcome the aforementioned challenges, we propose a general deep learning framework, called *FG-Net*, for large-scale point cloud understanding. We leverage deformable convolution for modeling the geometric structure, and pointwise attentional aggregation (AG) for mining the correlated features among point clouds. It should be noted that the deformable convolutional modeling can effectively adapt to the local geometry of objects by deformed kernels that dynamically adapt to diverse local geometries, while the correlated feature mining can capture the distributed contextual information in spatial locations and semantic features adaptively across a long spatial range. The modules in our network can be implemented with simple pointwise matrix multiplication and add operations, which can be easily parallelized by GPU for acceleration. As shown in Fig. 1, our method outperforms state-of-the-art (SOTA) ones in terms of both accuracy and efficiency, rendering it achievable to realize real-time perception performance on the large-scale point clouds. In summary, our work makes the following contributions.

1) We propose pointwise correlated feature mining and geometric-aware modeling module for large-scale point cloud understanding. Furthermore, we interpret the effectiveness of our network by visualizing the complementary features captured by our network modules.
2) We propose a feature pyramid-based residual learning architecture to leverage patterns at different resolutions in a memory-efficient way. Extensive experiments on real-world challenging benchmarks demonstrated that our approaches outperform SOTA ones in terms of accuracy and efficiency.
3) We propose a novel fast noise and outliers removal method and a points down-sampling strategy for large-scale point clouds, which simultaneously enhances the performance and improve the efficiency of semantic understanding tasks in the large-scale real-world scenes.

II. Related Works

Advanced deep learning techniques for images have been investigated extensively, and resulted in stunning performance [14]–[19]. Naturally, learning techniques have been exploited for point cloud processing and understanding, and the published works can be roughly categorized into voxel-based, projection-based, and point-based methods. The voxel-based and projection-based methods transform point clouds into different representations while the point-based methods process point clouds directly. These methods are mainly designed and tested on the relative small-scale point clouds of less than 10^5 points with block partitioning. Directly extending them to deal with large-scale point clouds will result in prohibitively expensive computational costs. Here, we discuss these methods thoroughly of their advantages and shortcoming, and the rationale that motivates our proposed framework.

A. Voxel-Based and Projection-Based Methods for Point Cloud Understanding

The most recent and typical voxel-based methods are SparseConv [11] and Minkowski CNN [12]. The voxel-based methods use 3-D convolutions which are intuitive extensions of 2-D counterparts. However, the computation cost and memory consumption of the voxel-based models increase cubically with the resolution of input point clouds. By contrast, the geometric information loss will be significant if we decrease the resolution of voxelization. Hence, it is quite hard to achieve real-time performance while considering the balance between accuracy and computational cost. The projection is also used to project point clouds into range images [20], [21] or multiview images [22], [23], to facilitate the use of 2-D CNNs. However, such projection inevitably leads to the loss of geometrical information. In practice of dealing with large-scale point clouds, the drawbacks of voxel-based and projection-based methods become more prohibitive.

B. Point-Based Methods for Point Cloud Understanding

The pointNet [24] is the pioneering work that extracts the pointwise feature directly using shared multilayer perceptron (MLP). The pointNet++ [9] extracts the local features using pointNet and considers local geometric relationships with the hierarchical grouping and abstraction as well as the

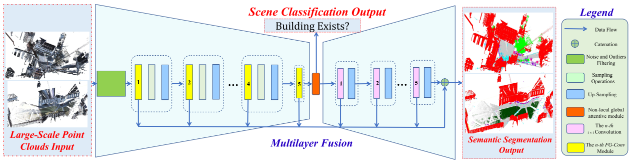

Scene Classification Output
Building Exists?
Large-Scale Point Clouds Input
1
2
4
5
1
2
5
Multilayer Fusion
Semantic Segmentation Output
Legend
Data Flow
Catenation
Noise and Outliers Filtering
Sampling Operations
Up-Sampling
Non-local global attentive module
The n-th 1×1 Convolution
The n-th FG-Conv Module

Fig. 2. Overall system framework of proposed *FG-Net*, and the proposed core module *FG-Conv* (shown in bright yellow) can be integrated into *FG-Net* with feature pyramid-based multiresolution residual learning.

multiscale and resolution grouping. More point-based methods [13], [25], [26] have been proposed recently with complicated network design to aggregate local features. However, all these methods are not able to model intrinsic geometric structures of points or to capture the nonlocal distributed contextual correlations in spatial locations and semantic features effectively. There are also a series of new explorations on how to implement convolution on point clouds. The methods [27]–[29] focus on how to learn kernels which can better capture the local geometry of points. However, the proposed convolutional kernels are too time consuming to be directly applied to deep neural networks for large-scale point cloud understanding [30]–[32]. Motivated by the challenges above, we proposed a novel lightweight point-based method to consider distributed long-range dependencies and learn kernels to capture the local structures of point clouds.

C. Efficient Large-Scale Point Cloud Understanding

It is till recently that more attention has been paid to efficient large-scale point cloud understanding. Previously, block partitioning [9], [24], [33] was utilized to divide large-scale point clouds into 1 m × 1 m sub-blocks before fed to networks. However, such an operation of partition is time-consuming and damages the spatial geometric contextual information among the objects of large-scale scene. Although several attempts [34], [35] have been made on large-scale point cloud segmentation, there are still some major problems existing: first, the FPS utilized by most of the previous methods require large computational cost which increases quadratically [1] with respect to number of input points N. Second, block partitioning causes that large-scale point cloud semantics cannot be inferred within one scan, which limits the volume of point clouds that can be processed. Some methods [34], [36] also try to combine voxelwise features with pointwise features to improve the performance. Analogous to the super-pixel conception for images, the super-point [13] method in point clouds is also introduced to apply graph convolutions on large-scale points. But due to the high computational cost of voxelization or graph construction, it can hardly achieve real-time performance.

III. Proposed Methodology

In this section, a fast deep learning method leveraging correlated feature mining and geometric-aware modeling is proposed for large-scale point cloud understanding. As illustrated in Fig. 2, our *FG-Net* takes raw point clouds of a large-scale complex scene as input and gives the predictions of object classification and semantic segmentation simultaneously. The details are given as follows.

A. Proposed Network Architecture for Large-Scale Point Cloud Understanding

We design the network module *FG-Conv* to capture the feature correlations and model the local geometry of point clouds simultaneously. Leveraging feature pyramid-based residual learning, *FG-Conv* can be integrated any point-based network as the core module for large-scale point cloud understanding. As shown in Fig. 3, the core network module *FG-Conv* includes three components: 1) pointwise-correlated feature mining (PFM); 2) geometric convolutional modeling (GCM); and 3) AG, which are detailed as follows.

1) Pointwise Correlated Features Mining: The point clouds after filtering are represented as *x-y-z* coordinates with features. The features can consist of raw RGB, surface normal information, intensity of point clouds, and even learned latent features. In fact, our method supports any kind of 3-D Data even if only 3-D coordinate can be obtained including RGB-D data, because it only requires the position information while others are optional. Denote the full input point clouds as the matrix $\boldsymbol{P} \in \mathbb{R}^{N\times(3+f^{\text{in}})}$, where N is the number of points and f^{in} is the dimension of input features, respectively. The ith vector in $\boldsymbol{P}$ can be denoted as $\boldsymbol{p_i} = (\boldsymbol{x_i}, \boldsymbol{f_i})^T$, where $\boldsymbol{x_i} \in \mathbb{R}^3$, $\boldsymbol{f_i} \in \mathbb{R}^{f^{\text{in}}}$, $i = 1, 2, 3, \ldots, N$. For the point $\boldsymbol{p_i}$, denote the kth point vector in the spherical neighborhood $B_r = \{\boldsymbol{s} \in \mathbb{R}^3, \|\boldsymbol{s} - \boldsymbol{x_i}\| \leq r\}$ as $\boldsymbol{p_k} = (\boldsymbol{x_k}, \boldsymbol{f_k})^T$, $\boldsymbol{x_k} \in \mathbb{R}^3$, $\boldsymbol{f_k} \in \mathbb{R}^{f^{\text{in}}}$, $k = 1, 2, 3, \ldots, K$, in which r is the radius of the neighborhood. The similarity score g_k which is the inner product of $\boldsymbol{p_k}$ and $\boldsymbol{p_i}$ is calculated as

$$g_k = \frac{\boldsymbol{p_k^T p_i}}{\|\boldsymbol{p_k}\| \|\boldsymbol{p_i}\|}. \tag{1}$$

It gives a good evaluation of the similarity of neighboring points in spatial locations and features. For each of the K neighboring points, g_k can be calculated and they constitute a similarity score vector $\boldsymbol{r_k} \in \mathbb{R}^K, k = 1, 2, 3, \ldots, K$. However, the similarity scores are not relevant to any specific task such as classification or segmentation. Thus, the attentional technique is introduced to make the new similarity score $\boldsymbol{z_k}$ adaptive to a specific task by training of deep networks

$$\boldsymbol{z_k} = \sigma(\boldsymbol{w_1 r_k}), \boldsymbol{z_k} \in \mathbb{R}^K \tag{2}$$

where $\boldsymbol{w_1} \in \mathbb{R}^{K\times K}$ is the weight matrix to be learned. σ is the softmax function to normalize the attentional weights. All $\boldsymbol{p_k}$ constitute the matrix $\boldsymbol{P_k} = (\boldsymbol{p_1}, \boldsymbol{p_2}, \boldsymbol{p_3}, \ldots, \boldsymbol{p_k})^T$, $\boldsymbol{P_k} \in \mathbb{R}^{K\times(3+f^{\text{in}})}$. Then, each element of $\boldsymbol{z_k}$ is multiplied with each row of $\boldsymbol{P_k}$ through element-to-row multiplication to obtain the augmented attentional feature matrix $\boldsymbol{P'}_k \in \mathbb{R}^{K\times(3+f^{\text{in}})}$. From now on, the augmented feature (e.g., $\boldsymbol{P'}_k$) which encodes both geometry and feature correlations are called feature for the sake of brevity. Next, $\boldsymbol{P'}_k$ is concatenated with their corresponding input feature $\boldsymbol{P_k}$ to obtain the enhanced feature $\boldsymbol{F_k^1} \in \mathbb{R}^{K\times f^{\text{mid}}}$ ($f^{\text{mid}} = 6 + 2f^{\text{in}}$). In this way, the local contextual relationship can be captured, and the similarity of features

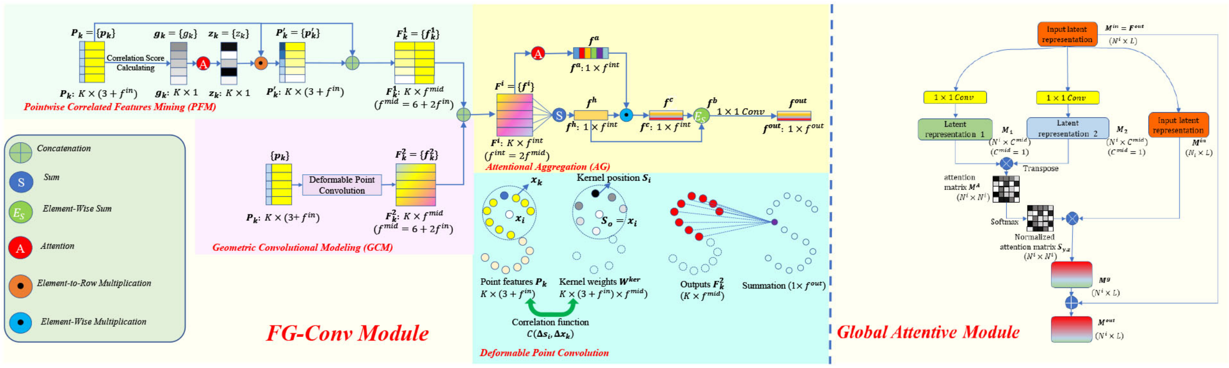


Fig. 3. Detailed illustration of our proposed novel pointwise-correlated feature and GCM module (*FG-Conv Module*). The deformable convolution operation is illustrated at the right bottom corner. The query point is denoted as x_i and the kth neighbor point denoted as x_k, and the output vector of point convolution is the dot product of point features and kernel weights. The feature vector is summed in AG. On the right side of the dash line is detailed illustration of the global attentive module that functions as "Global feature Extraction" in Fig. 4. The feature representations $\boldsymbol{M}^{\mathbf{in}}$ and $\boldsymbol{M}^{\mathbf{out}}$ in this module are corresponding to Fig. 4(b).

is enhanced adaptively and selectively by the attentive weighting in a learnable way. The similar feature elements in latent space are enhanced while distinct ones are attenuated.

2) Geometric Convolutional Modeling: After the PFM, the local correlated features can be largely captured, but the geometric structure of points cannot be sufficiently modeled. Inspired by the great success of deformable convolution in the image recognition [37], we extend deformable convolutions from image to point clouds to model the irregular and unordered 3-D structures. Similar to 2-D deformable convolutions, the deformable 3-D kernels in Euclidean space are a set of learnable points that conform to the local structures of point clouds, thus, the dominant local geometric shapes of the points can be activated by the corresponding kernels in the neighborhood. Note that kernel deformations can adapt to the local geometry of points in a learnable way by elaborately designed optimization functions. As shown in the right bottom of Fig. 3, like convolutional neural networks in image processing, the convolution on points is defined as

$$\boldsymbol{F}^2_{\boldsymbol{k}}(\boldsymbol{p_k}, \boldsymbol{p_i}) = \sum_{\boldsymbol{p_k} \in B_r} \boldsymbol{K}(\boldsymbol{p_k}, \boldsymbol{p_i})\boldsymbol{p_k}. \tag{3}$$

The core problem is that point clouds are unstructured and unordered, which makes it difficult for point convolutional kernel function $\boldsymbol{K}(\boldsymbol{p_k}, \boldsymbol{p_i})$ to learn representative local geometric patterns. We design the correlation function to measure the correspondence between kernel points and local geometry. To be more specific, the closer the kernel points to the input point, the higher the correlation value should be assigned. Denote the difference between $\boldsymbol{x_k}$ and $\boldsymbol{x_i}$ as $\Delta\boldsymbol{x_k} = \boldsymbol{x_k} - \boldsymbol{x_i}$. The N^s pseudo kernel points $\boldsymbol{S_i} \subset B_r$ centered at $\boldsymbol{S_o}$ ($\boldsymbol{S_o} = \boldsymbol{x_i}$) are designed so as to imitate the convolutional kernels in image processing, ($i = 1, 2, 3, \ldots, N^s$). The relative coordinates of pseudo kernel points $\boldsymbol{S_i}$ and the center point $\boldsymbol{S_o}$ are given as: $\Delta\boldsymbol{s_i} = \boldsymbol{S_i} - \boldsymbol{S_o}$. The correlation function can be learned in an end-to-end way, which is formulated as

$$C(\Delta\boldsymbol{s_i}, \Delta\boldsymbol{x_k}) = \frac{1}{\|N^s\|}\exp\left(-\frac{\|\Delta\boldsymbol{s_i} - \Delta\boldsymbol{x_k}\|^2}{m\lambda^2}\right) \tag{4}$$

where N^s is the number of kernel points, m is a constant, and λ is the parameter determining the influence distance of kernel points. Then, the kernel function is given as the sum of all relations with learnable weights as shown in the following:

$$K(\boldsymbol{p_k}, \boldsymbol{p_i}) = \sum_{n=1}^{N^s} C(\Delta\boldsymbol{s_i}, \Delta\boldsymbol{x_k})\,\boldsymbol{W}^{\text{ker}} \tag{5}$$

where $\boldsymbol{W}^{\text{ker}} \in \mathbb{R}^{(3+f^{\text{in}})\times f^{\text{mid}}}$ is the weight matrix of MLP layers, and $3+f^{\text{in}}$ and f^{mid} are input and output channel numbers, respectively. The $C(\Delta\boldsymbol{s_i}, \Delta\boldsymbol{x_k})$ is the kernel assembling functions that should be learned based on the distance between the kernel positions and point positions, that is, $\|\Delta\boldsymbol{s_i} - \Delta\boldsymbol{x_k}\|$. Different from the linear correlation function proposed in the KPConv [28] which may not be optimal in evaluating the weight of assembling, we have further tested the assembling function design of the square assembling function, Gaussian assembling function, and the designed learnable assembling functions with learnable weights. During the optimization process, the kernel points are forced to adapt to the dominant structures in the local point clouds. Finally, the feature after deformable convolution $\boldsymbol{F}^2_{\boldsymbol{k}} \in \mathbb{R}^{K\times f^{\text{mid}}}$ ($f^{\text{mid}} = 6 + 2f^{\text{in}}$) can be obtained. In this way, the local geometric structures are well captured and the dominant structural features are enhanced.

3) Attentional Aggregation: The attention mechanism is utilized to leverage the feature level and geometric level patterns without large information loss. As shown in Fig. 3, the integrated neighboring features can be represented as $\boldsymbol{F^i} \in \mathbb{R}^{K\times f^{\text{int}}}$ ($f^{\text{int}} = 2f^{\text{mid}}$). Then, the attentive score for aggregation is defined as $\boldsymbol{w_2} \in \mathbb{R}^K$, which will adaptively learn the importance of each feature. The weighted attentional feature $\boldsymbol{f^a} \in \mathbb{R}^{f^{\text{int}}}$ can be given as

$$\boldsymbol{f^a} = \sigma\left(\boldsymbol{w_2}\,\boldsymbol{f^i}\right). \tag{6}$$

The summed feature can be given as: $\boldsymbol{f^h} = \sum_{i=1}^{K}\boldsymbol{f^i}, \boldsymbol{f^h} \in \mathbb{R}^{f^{\text{int}}}$. Then, the elementwise multiplication between $\boldsymbol{f^a}$ and $\boldsymbol{f^h}$ is utilized to obtain the learned feature $\boldsymbol{f^c} = \boldsymbol{f^a} \odot \boldsymbol{f^h}$. The final features $\boldsymbol{f^b}$ is the sum of original feature and learned feature: $\boldsymbol{f^b} = \boldsymbol{f^h} + \boldsymbol{f^c}$. We apply the MLP layer to control the

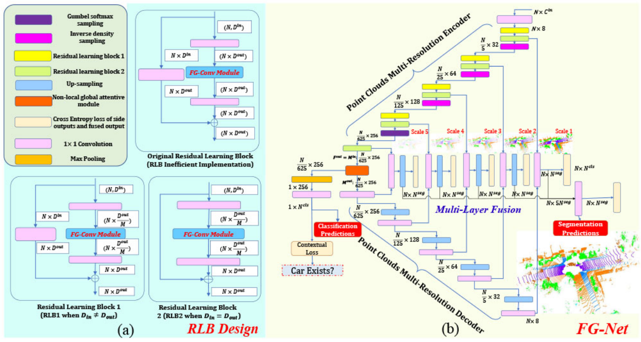

Gumbel softmax sampling
Inverse density sampling
Residual learning block 1
Residual learning block 2
Up-sampling
Non-local global attentive module
Cross Entropy loss of side outputs and fused output
1× 1 Convolution
Max Pooling
(N, D^{in})
$N \times D^{in}$
$(N \times D^{out})$
FG-Conv Module
$N \times D^{out}$
Original Residual Learning Block (RLB Inefficient Implementation)
$(N \times \frac{D^{out}}{M})$
Residual Learning Block 1 (RLB1 when $D_{in} \neq D_{out}$)
Residual Learning Block 2 (RLB2 when $D_{in} = D_{out}$)
(a)
RLB Design
Point Clouds Multi-Resolution Encoder
$N \times C^{in}$
$N \times 8$
$\frac{N}{5} \times 32$
$\frac{N}{25} \times 64$
$\frac{N}{125} \times 128$
$\frac{N}{625} \times 256$
Scale 5
Scale 4
Scale 3
Scale 2
Scale 1
$F^{out} = M^{in}$: $\frac{N}{625} \times 256$
M^{out}: $\frac{N}{625} \times 256$
1×256
$1 \times N^{cls}$
Classification Predictions
Contextual Loss
Car Exists?
$N \times N^{seg}$
$N \times N^{cls}$
$N \times 5N^{seg}$
Multi-Layer Fusion
Segmentation Predictions
Point Clouds Multi-Resolution Decoder
(b)
FG-Net

Fig. 4. (a) Detailed RLB design of our proposed *FG-Net*, in which we show the original inefficient RLB and our designed RLB, respectively. (b) Proposed feature pyramid residual learning network. N^{cls} and N^{seg} stand for the number of classes in classification of the presence of objects or not, and the number of classes in semantic segmentation, respectively.

dimension of the output vector flexibly and give the meaningful aggregated feature $\boldsymbol{f}^{\text{out}} \in \mathbb{R}^{f^{\text{out}}}$ containing both local correlated features and enhanced local geometry.

4) Feature Pyramid Hierarchical Residual Architecture: The Resnet [38]-based architecture has achieved great success in image recognition. Motivated by the residual learning paradigm, we proposed a general deep network that is specially designed for classification and semantic segmentation of large-scale point clouds. To the best of our knowledge, until recently, there are several methods [39], [40] starting to use residual learning for point cloud recognition, but their attempts are limited to small-scale point clouds. Thus, the fitting capacity of residual architecture cannot be fully demonstrated. We propose a feature pyramid-based multiscale fusion strategy for adaptively aggregating features from different layers of the network. Leveraging a deep residual structure, memory-efficient deep networks can be built.

As shown in Fig. 4, the encoder–decoder-based network structure can be utilized to obtain point clouds at multiple resolutions. In image processing, the networks are supposed to extract large feature maps for small objects and small feature maps for big objects [41], [42]. It should be noted that scale variation in images will not exist in 3-D point clouds. Different from images in which the scale of objects will vary with the distance, the scale of point clouds will keep constant. Hence, the deconvolution by interpolation must be conducted to recover the points to the original resolution. As shown in Fig. 4(a), in the residual learning block (RLB), denote the input dimension and output dimension of RLB as D^{in} and D^{out}, respectively. Unlike some deep architectures which are memory consuming, we reduce the feature dimension in the original RLB to D^{out}/M ($M = 8$ is utilized in our framework) by 1×1 convolution before feeding them into *FG-Conv* module, which reduce the parameters by 9.6 times. The accuracy for classification and segmentation can also be maintained through residual learning, which will be given in experiments. Another 1×1 convolution will be applied to recover the feature dimension. At the block connecting two stages, 1×1 convolution should be applied in skip link for increasing the feature dimensions. Then, the global feature extraction in Section III-A5 will be conducted to obtain the latent global features $\boldsymbol{M}^{\textbf{out}}$ from $\boldsymbol{M}^{\text{in}}$, which can be directly utilized for classification predictions. The point clouds are all upsampled after the h ($h = 5$ in our case) convolutional blocks. Unlike previous methods which directly used the upsampled features for segmentation, we propose to fuse the predictions at different resolutions and use the supervised loss to guide the training process. It turned out the hierarchical structure will give better results for pointwise large-scale point cloud segmentation.

5) Point Cloud Global Feature Extraction: The global and long-range dependencies in point clouds should also be captured before doing upsampling and giving the pointwise predictions. Due to the limited receptive field of the neural layer mentioned above, the global contextual semantic patterns cannot be fully obtained. We leverage the self-attentional module shown in the right side of the dash line in Fig. 3 to selectively enhance the closely relevant elements in the global feature $\boldsymbol{M}^{\textbf{in}}$. After the global relationship mining by this module, both the local and global relationships in features and geometry will be captured adaptively. Then, the feature representation with combined local or nonlocal semantic contextual correlations will be adaptively obtained to facilitate the subsequent recognition task. Given the original local feature map $\boldsymbol{F}^{\textbf{out}} = \boldsymbol{M}^{\textbf{in}} \in \mathbb{R}^{N^i \times L}$, $(N^i = (N/625), L = 256$ in our case) as shown in Figs. 3 and 4, the 1×1 convolution with weight $\boldsymbol{W}^{\boldsymbol{G}} \in L \times C^{\text{mid}}$ ($C^{\text{mid}} = 1$ in our case) is used to transform the feature map into latent representations $\boldsymbol{M_1}$ and $\boldsymbol{M_2}$ for further obtaining the similarity of each two elements in $\boldsymbol{F}^{\textbf{out}}$. After $\boldsymbol{M_1}$ and $\boldsymbol{M_2}$ are obtained, the dot product between them can be conducted to obtain the relevant score matrix $\boldsymbol{M}^{A} \in \mathbb{R}^{N^i \times N^i}$ which is given as

$$\boldsymbol{M}^{\boldsymbol{A}} = \boldsymbol{M_1}\boldsymbol{M_2}^{T}. \tag{7}$$

Each element $m_{y,z}$ in $\boldsymbol{M}^{\boldsymbol{A}}$ gives the relevance score between the representation $\boldsymbol{M_1}$ and $\boldsymbol{M_2}$. Then, the softmax is applied to normalize the latent attentional scores to obtain the final self-relation weights $\boldsymbol{S_{y,z}} \in \mathbb{R}^{N^i \times N^i}$ of the latent representation M^{in}. Each element $s_{y,z}$ of $\boldsymbol{S_{y,z}}$ can be represented as

$$s_{y,z} = \frac{\exp(m_{y,z})}{\sum_{y=1}^{N^i}\sum_{z=1}^{N^i}\exp(m_{y,z})}. \tag{8}$$

The attention weights $s_{y,z}$ reveal the correlations among all local and global features. The more related distributed feature relationships, even in the nonlocal region, can be effectively captured, and larger attention weights are assigned in $s_{y,z}$ to enhances their similar semantic contexts. Finally, the attention scores are applied to all elements in $\boldsymbol{M}^{\textbf{in}}$ to produce the global attentional vector $\boldsymbol{M}^{\boldsymbol{g}}$

$$\boldsymbol{M}^{\boldsymbol{g}} = \boldsymbol{S_{y,\,z}}\boldsymbol{M}^{\textbf{in}}. \tag{9}$$

The global attentional vector will provide higher weights to activate correlated significant features, and will provide lower weights to suppress the irrelevant less important features. The consolidated feature $\boldsymbol{M}^{\textbf{out}}$ is the sum of $\boldsymbol{M}^{\boldsymbol{g}}$ and $\boldsymbol{M}^{\textbf{in}}$, which is given as: $\boldsymbol{M}^{\textbf{out}} = \boldsymbol{M}^{\boldsymbol{g}} + \boldsymbol{M}^{\textbf{in}}$.

Ultimately, the global contextual representation $\boldsymbol{M}^{\boldsymbol{g}}$ is fused with the local aggregated representation $\boldsymbol{M}^{\textbf{in}}$ for a comprehensive encoding of local and global correlated features. The predictions of classification can be directly obtained from the aggregated latent features $\boldsymbol{M}^{\mathrm{out}}$ and the segmentation results can also be learned by up-sampling. As shown in our ablation studies, the point cloud global feature correlations have a boost on the segmentation performance because the nonlocal long range correlations of point clouds can be effectively captured. Finally, the global attentive function serves as a good module for global correlation mining. It is also complements with the pointwise-correlated feature mining.

6) Optimization Function Formulation and Data Augmentation: As mentioned in Section III-A2, denote the relative coordinates of kernel points as $\boldsymbol{s_i}$, $i = 1, 2, \ldots, N^s$ and the learned deformation as $\Delta\boldsymbol{s_i}$. The losses utilized for the deformable convolution are designed as

$$L_{\mathrm{fit}}(\Delta s_i) = \sum_{i=1}^{N^s} \min_{\Delta s_i}\left(\frac{\|\Delta \boldsymbol{x_k} - (\boldsymbol{s_i} + \Delta \boldsymbol{s_i})\|}{m\sigma^2}\right)^2 \tag{10}$$

which is utilized to match the kernel positions with local geometries of point clouds

$$L_{\mathrm{rep}}(\Delta \boldsymbol{s_i}) = \min_{\Delta s_i} \sum_{i=1}^{N^s}\sum_{j=1}^{N^s} \frac{1}{\|\boldsymbol{s_i} + \Delta \boldsymbol{s_i} - \boldsymbol{s_j} - \Delta \boldsymbol{s_j}\|} \tag{11}$$

which is the repulsive loss utilized to keep distance between different kernels

$$L_{\mathrm{att}}(\Delta \boldsymbol{s_i}) = \min_{\Delta s_i} \sum_{i=1}^{N^s} \|\boldsymbol{s_i} + \Delta \boldsymbol{s_i}\|^2 \tag{12}$$

which is to keep the kernel points from diverging and make them inside the query ball. The kernel loss will be the sum of above three losses. That is, $L_{\mathrm{ker}}(\Delta \boldsymbol{s_i}) = L_{\mathrm{fit}}(\Delta \boldsymbol{s_i}) + L_{\mathrm{rep1}}(\Delta \boldsymbol{s_i}) + L_{\mathrm{rep2}}(\Delta \boldsymbol{s_i})$. As shown in Fig. 4, the losses at different stages of the network are also summed, which can be formulated as the cross-entropy loss denoted as L_1

$$\begin{aligned} L_1(\boldsymbol{W}) = {} & \sum_{n=1}^{N}\sum_{h=1}^{H} \alpha^{(h)} \hat{y}_i \log\left(P_{\mathrm{seg}}\left(\boldsymbol{p}_{\boldsymbol{i}}^{(h)}, \boldsymbol{W}\right)\right) \\ & + \beta \hat{y}_i^{\mathrm{fuse}} \log\left(P_{\mathrm{seg}}\left(\boldsymbol{p}_{\boldsymbol{i}}^{(\mathrm{fused})}, \boldsymbol{W}\right)\right). \end{aligned} \tag{13}$$

$\alpha^{(h)}$ denotes the weight at stage h of the residual network, $\boldsymbol{W}$ denotes the weight of the entire network, $\boldsymbol{p}_{\boldsymbol{i}}^{(h)}$ denotes the upsampled point clouds at stage h, $\boldsymbol{p}_{\boldsymbol{i}}^{(\mathrm{fused})}$ denotes the fused point clouds, and $\hat{y}_i$ and $\hat{y}_i^{\mathrm{fuse}}$ denote the segmentation ground truth of points at different stages and fused points, respectively. P_{seg} denotes the segmentation prediction of the networks. We also propose to use the contextual loss shown in Fig. 4 to predict the presence of objects or not in the scene to consider semantic contexts of the scene, which can be given as

$$L_2(\boldsymbol{W}) = -\sum_{i=1}^{I} \hat{y}_i^{\mathrm{pre}} \log(P_{\mathrm{cls}}(\boldsymbol{p_i}, \boldsymbol{W})) \tag{14}$$

where $\hat{y}_i^{\mathrm{pre}}$ indicates whether the object presents in the scene or not and P_{cls} is the classification prediction. This loss equally considers all the semantic categories appearing in the scene. The total loss of the network is: $L_c(\boldsymbol{W}, \Delta \boldsymbol{s_i}) = L_1(\boldsymbol{W}) + L_2(\boldsymbol{W}) + L_{\mathrm{ker}}(\Delta \boldsymbol{s_i})$. The kernel positions and network parameters are jointly optimized in an end-to-end manner.

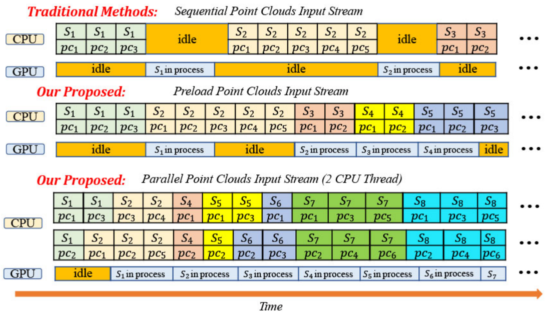

Traditional Methods: Sequential Point Clouds Input Stream
CPU
GPU
S_1 S_1 S_1 / pc_1 pc_2 pc_3
idle
S_2 S_2 S_2 S_2 S_2 / pc_1 pc_2 pc_3 pc_4 pc_5
idle
S_3 S_3 / pc_1 pc_2
idle
S_1 in process
idle
S_2 in process
idle
Our Proposed: Preload Point Clouds Input Stream
CPU
GPU
S_1 S_1 S_1 S_2 S_2 S_2 S_2 S_2 S_3 S_3 S_4 S_4 S_5 S_5 S_5
pc_1 pc_2 pc_3 pc_1 pc_2 pc_3 pc_4 pc_5 pc_1 pc_2 pc_1 pc_2 pc_1 pc_2 pc_3
idle
S_1 in process
idle
S_2 in process
S_3 in process
S_4 in process
idle
Our Proposed: Parallel Point Clouds Input Stream (2 CPU Thread)
CPU
S_1 S_1 S_2 S_2 S_4 S_5 S_5 S_6 S_7 S_7 S_7 S_8 S_8 S_8
pc_1 pc_3 pc_3 pc_4 pc_1 pc_1 pc_3 pc_1 pc_1 pc_3 pc_5 pc_1 pc_3 pc_5
S_1 S_2 S_2 S_2 S_4 S_5 S_6 S_6 S_7 S_7 S_7 S_8 S_8 S_8
pc_2 pc_1 pc_2 pc_5 pc_2 pc_2 pc_2 pc_3 pc_2 pc_4 pc_6 pc_2 pc_4 pc_6
GPU
idle
S_1 in process
S_2 in process
S_3 in process
S_4 in process
S_5 in process
S_6 in process
S_7
Time

Fig. 5. Proposed multithread CPU parallel computation for point cloud streams. The noise and outliers filtering is done on CPU while sampling and deep network processing are done on GPU. S_i stands for *ith* CPU processing stream, pc_j stands for the *j*th point cloud batch in a single CPU stream.

B. Noise and Outliers Filtering

The noise and outliers can be removed very effectively with a speed of 0.61s per million (10^6) points due to the *Octree-based* fast nearest-neighbor search, similar to the implementation of [43]. It should be noted that the noise and outliers filtering is only utilized during the training of the network. When testing, we directly test the methods on the testing set of each benchmark. Our simple but effective method can be utilized to remove noise and outliers of points while enhancing the performance of point cloud understanding in the meanwhile. The implementation details for acceleration of our framework is introduced in Section III-C.

C. Implementation Details for Acceleration of Our Framework

1) IGSAM for Fast Learning-Based Sampling: The sampling methods play a very significant role in processing point clouds by convolutional neural networks. To tackle the large computational overhead when processing millions of point clouds, we propose efficient sampling methods *IGSAM* to achieve fast and effective point cloud understanding. We design a novel learning-based gumbel softmax sampling (GSS) which adaptively selects the significant points based on optimization objectives. By integrating it with inverse density sampling (IDS), points can be sampled efficiently with density awareness while the important features for point cloud understanding is maintained.

2) Acceleration by Multithread Parallel Computation on CPU: The noise and outliers filtering is implemented on CPU while deep network computations are implemented on GPU. The CPU is a Intel Core i7-8700T Processor with frequency of 2.40 GHz. It has six cores and 12 threads and we have used full six cores for computation. It should be noted that we have reused the radius-based ball query in both point cloud filtering and network operations for accelerations. As shown in Fig. 5, we have preloaded the next stream of points to CPU before the network computation on GPU is finished for acceleration. The multithread data loading and computation on CPU is utilized to accelerate the *Octree-based*

TABLE I
COMPARISON OF SEGMENTATION ON S3DIS (S3) AND SHAPENETPART (SP) WITH (W/) AND WITHOUT (W/O) FILTERING

Method	S3 (w/o)	S3 (w/)	SP (w/o)	SP (w/)
SPG [13]	62.1	63.2	76.1	80.2
Shellnet [44]	66.8	67.9	73.2	80.1
PointCNN [33]	65.4	65.8	79.1	83.4
Kpconv [28]	67.1	68.5	83.2	85.1
DGCNN [45]	56.1	58.2	70.3	76.8
FKA-Conv [4]	68.1	68.6	80.5	84.1
FG-Net (Ours)	**70.2**	**70.8**	**86.2**	**87.7**

TABLE II
COMPARISONS OF DIVERSE ASSEMBLING FUNCTION ON SEGMENTATION PERFORMANCE

Assembling Function	S3DIS	Scannet	NPM3D	ShapeNetPart
Linear Assembling	68.2	63.2	79.6	86.9
Square Assembling	68.9	63.2	79.2	86.1
Gaussian Assembling	68.5	67.9	76.4	86.5
Learnable Assembling	**70.8**	**69.0**	**81.9**	**87.7**

query process, which reduces idle periods notably in subsequent computations both on CPU and GPU, as also shown in Fig. 5.

IV. EXPERIMENTAL RESULTS

A. Experimental Setup

To demonstrate the effectiveness of our method, we have experimented extensively on eight large-scale point cloud understanding benchmarks. We implemented the network in Tensorflow and optimized it with Adam optimizer and initial learning rate of $1e^{-4}$. The batch size equals 4 during training and equals 8 during validation and testing for all tested benchmarks. Also, the point clouds are randomly rotated around each axis x, y, z with an angle $\phi \in [0, 2\pi]$. The scaling is also applied along x, y, z axis with a scalar $\mu \in [0.85, 1.15]$ for data augmentation. The network is trained and tested in parallel with 5×10^5 point clouds in each stream. The experiments are conducted on Nvidia GTX 1080 with 8-GB memory.

B. Experiments of Filtering and Sampling Methods, and With Diverse Assembling Functions

1) Noise and Outliers Filtering and Comparisons of Diverse Assembling Functions: We have tested the influence of proposed noise and outliers filtering on the semantic segmentation performance of S3DIS. The noise outlier filtering is only utilized in training. During testing, we directly use the trained model to do inference on point clouds with noise. Noted that we utilize 6-fold cross-validation on S3DIS to guarantee the generality and robustness. The noise filtering results with mean intersection over unions (mIOUs) are shown in Table I, it demonstrates that filtering has a boost on segmentation performance for diverse point cloud understanding methods. After removing unrelated isolated noise points, the meaningful semantics of point clouds is retained, and it boosts the performance of segmentation. As shown in Table II, the assembling function containing learnable weights has the best performance on most of the public real-scene benchmarks.

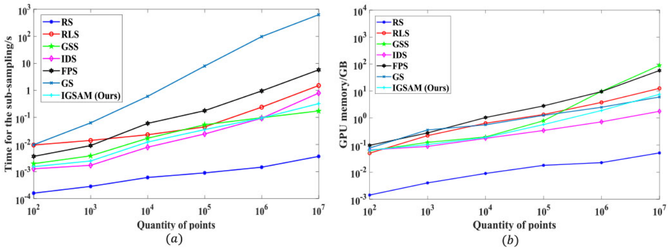


Fig. 6. Comparison of (a) computational cost and (b) memory consumption of different sampling methods.

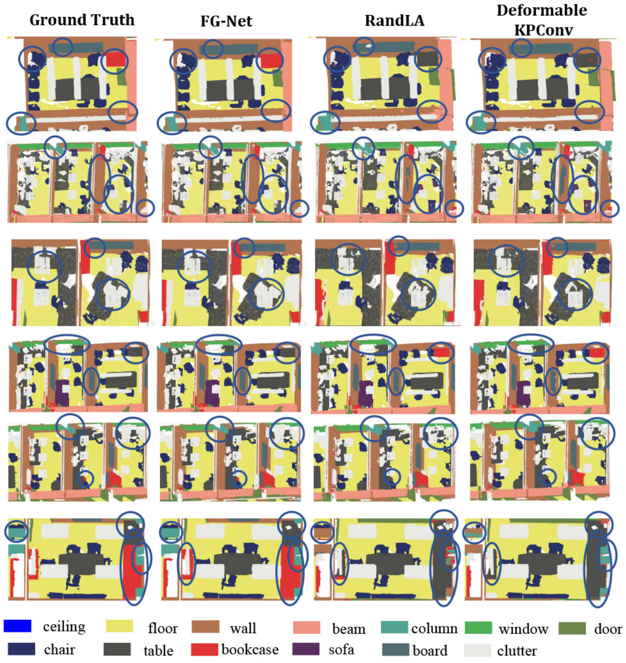


Fig. 7. Detailed comparisons of FG-Net with SOTA methods RandLA-Net and KPConv on S3DIS, the comparisons are highlighted in the blue circles.

2) Point Cloud Sampling Methods: To compare the efficiency our proposed *IGSAM* with different sampling methods, we have experimented their GPU memory usage and processing time on a single GTX 1080 GPU with 8-GB memory. The sampling methods include random sampling (RS), reinforcement learning-based sampling (RLS) [46], GSS [47], IDS, FPS, and generative network (GS) [48]-based Sampling. The point clouds are divided into batches consisting of 10^2, 10^3, 10^4, 10^5, 10^6, and 10^7 points, respectively, then the batches of points are downsampled five times which imitates the downsampling in our network shown in Fig. 4. The total time and memory consumption of sampling methods on different numbers of points are illustrated in Fig. 6. It can be demonstrated that RS has the fastest processing speed with the smallest memory consumption. However, RS will result in a stochastic loss in meaningful information, which will give unsatisfactory segmentation results. As shown in Table III, mIOUs will drop significantly from 70.8% to 66.8% if RS is adopted. It should also be noted that GSS is not suitable for more than 10^6 points because the GPU memory will increase greatly with the number of points. Hence, we only use GSS in the last layer of the network when the number of points is less

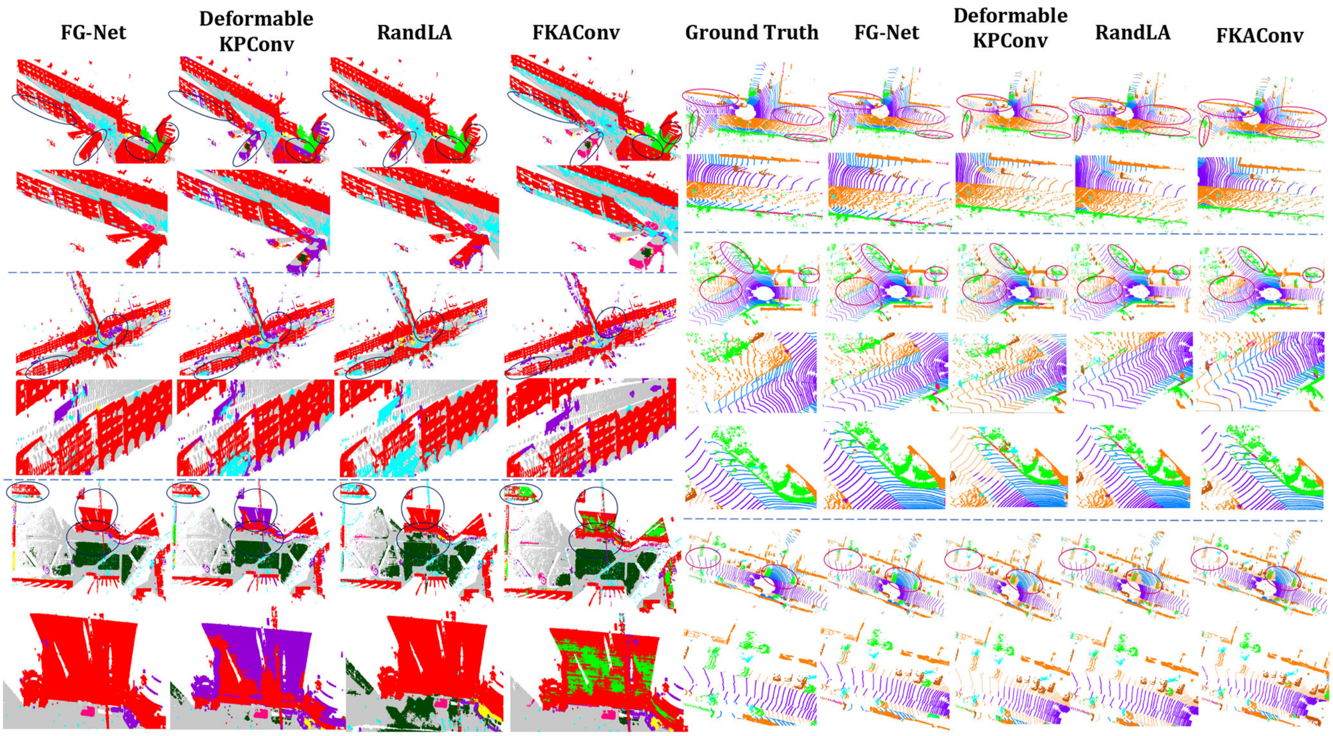


Fig. 8. Detailed Comparisons of FG-Net with SOTA methods RandLA-Net, Deformable KPConv, and FKA-Conv on large-scale point cloud segmentation benchmarks Semantic3D (left), and SemanticKITTI (right) with zoom-in results shown below. The ground truth for the Semantic3D test set is not publicly available. The different scenes are separated by dash lines. The comparisons are highlighted by circles.

TABLE III
Comparison of Segmentation Performance on S3DIS With Diverse Sampling Methods

Sampling Method	RS	RLS	***IGSAM***	IDS	FPS	GS
SPG [13]	61.7	62.2	63.2	62.6	61.8	62.5
Shellnet [44]	66.2	66.3	67.6	66.9	66.1	66.3
PointCNN [33]	64.9	65.3	66.8	66.1	65.9	65.3
KPConv [28]	66.5	67.1	67.5	67.3	66.7	66.3
DGCNN [45]	55.2	56.0	58.4	57.5	57.9	56.0
FKA-Conv [4]	64.6	65.2	68.6	66.2	66.1	65.3
FG-Net (Ours)	**66.8**	**70.3**	**70.8**	**70.3**	**69.5**	**69.9**

than 10^5. Leveraging IDS with adaptability to the local density of points, our *IGSAM* achieve the best performance among different sampling methods with only a marginal increase of computational cost compared with RS. The segmentation mIOUs using different sampling methods are also shown in Table III. It can be concluded that our sampling methods give the best performance among all sampling methods on the S3DIS benchmark with mIOUs of 70.8%, which demonstrates the effectiveness of our proposed sampling strategy.

C. Experiments of Large-Scale Scene Understanding

We have experimented our method extensively on nearly all existing large-scale point cloud understanding benchmarks, including ModelNet40 [52], ShapeNet-Part [53], PartNet [54], S3DIS [5], NPM3D [49], Semantic3D [55], Semantic-KITTI [50], and Scannet [56]. The outdoor datasets, such as NPM3D, Semantic3D, and Semantic-KITTI are mainly captured by LiDAR sensors, while the indoor datasets, such as S3DIS and Scannet are mainly obtained by RGB-D cameras and transformed into the representation of point clouds. The detailed comparisons of our method with current SOTA point cloud understanding methods in speed, accuracy, and memory are shown in Table IV. The qualitative experiments of large-scale real-world scene parsing are shown in Figs. 7, and 8, respectively. The mIOUs of 77.2%, 70.8%, 81.9%, and 58.2% are attained on Semantic3D [55], S3DIS [5], NPM3D [49] and challenging fine-grained part segmentation benchmark PartNet [54], respectively, with real-time performance of 18.6 Hz per LiDAR scan with 5 × 10^5 points, which outperforms SOTA methods in terms of accuracy, speed, and memory efficiency. As shown in Table IV, we achieve the best or the second best performance on public benchmarks with the least network running time, which demonstrates superior effectiveness, running speed, and memory efficiency of our proposed method. The transfer learning results shown in Fig. 9 also demonstrates our networks learn the underlying latent model of feature representations that generalizes well across new scenes.

D. Visualization of the Network Modules

1) Visualization of the Deformable Convolutional Kernels: To better demonstrate the geometry adaptive capacity of the deformable convolutions, the deformable kernel are visualized in Fig. 10. It can be seen that kernel points are adaptively deformed to capture different geometric structures in the original point clouds. Hence, in the test phase, the specific geometric structures in the unseen scene will be effectively captured and described by deformable kernels. In this way, we can model the geometry of the scene in a learnable

TABLE IV

COMPARISONS OF CLASSIFICATION OR SEGMENTATION PERFORMANCE, RUNNING TIME, AND CONSUMED MEMORY OF OUR METHOD WITH SOTA METHODS ON DIFFERENT LARGE-SCALE UNDERSTANDING BENCHMARKS. THE RED AND BLUE COLORS REPRESENT THE BEST, THE SECOND-BEST RESULTS, RESPECTIVELY. RESULTS ARE RETRIEVED FROM ONLINE BENCHMARKS AT JUNE 15, 2021, OR FROM ORIGINAL PAPERS OF SOTA METHODS

Benchmarks	ModelNet	ShapeNet	NPM3D	Semantic3D	PartNet	S3DIS	Scannet	Time(s) /10^5 points	Memory(M)
FG-Net (Ours)	**93.8**	**87.7**	**81.9**	**77.2**	**58.2**	**70.8**	**69.0**	**0.052**	**63.1**
FKA-Conv [5]	92.5	84.1	**82.7**	74.6	55.4	68.6	62.5	0.923	**263.8**
Defromable KPConv [49]	92.7	85.1	75.9	73.1	55.5	68.5	68.6	2.465	1237.6
Rigid KPConv [49]	92.9	85.0	72.3	74.3	52.8	65.4	65.7	2.235	1187.9
RandLA-Net [50]	90.1	83.1	78.5	**77.4**	51.6	70.0	64.5	**0.918**	302.1
PAConv [51]	**93.9**	84.6	69.8	71.2	45.8	66.9	62.5	1.921	876.9

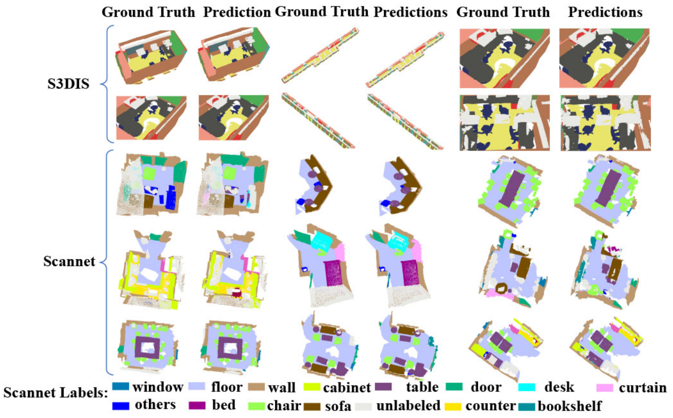


Fig. 9. Transfer learning results between S3DIS and Scannet. Please zoom in for details.

way to better enhance structural awareness in per-point-based processing.

2) Visualization of the Learned Features: In order to better demonstrate the geometry adaptive capacity of the deformable convolutions, the deformable kernel are visualized in Fig. 10. It can be seen that the kernel points are adaptively deformed to capture different kinds of geometric structure in the original point clouds. Therefore, in the test phase, the specific geometry structures in the unseen scene will be effectively captured and described by the deformable kernels.

3) Visualization of the Nonlocal Activation: Fig. 10 demonstrates that the nonlocal module captures the long-range dependencies of the same semantic category, such as chairs or bookcases. The contexts even far from each other can be nicely modeled and captured. It can also be observed that the nonlocal activation also provide rough results of segmentation prediction of the category of the query point, which is advantageous for further semantic segmentation tasks.

E. Efficiency and Online Performance

We have done runtime comparisons of our method compared with SOTA on the entire sequence of large-scale real-scene benchmark Semantic-KITTI [50]. The sequences are captured and fed into the networks at 25 Hz. The PaiSeg [57] is a recently developed method. Our method can reach 16.89, 19.53, 19.31, and 18.69 Hz for LIDAR scan 02, 04, 05, and 09, respectively. Compared with RSSP [1] and RandlaNet [35], the speed increase by 274% and 38.5% while the memory consumption reduce by 46.5% and 8.6%, respectively, which is a prominent progress in running speed and memory efficiency.

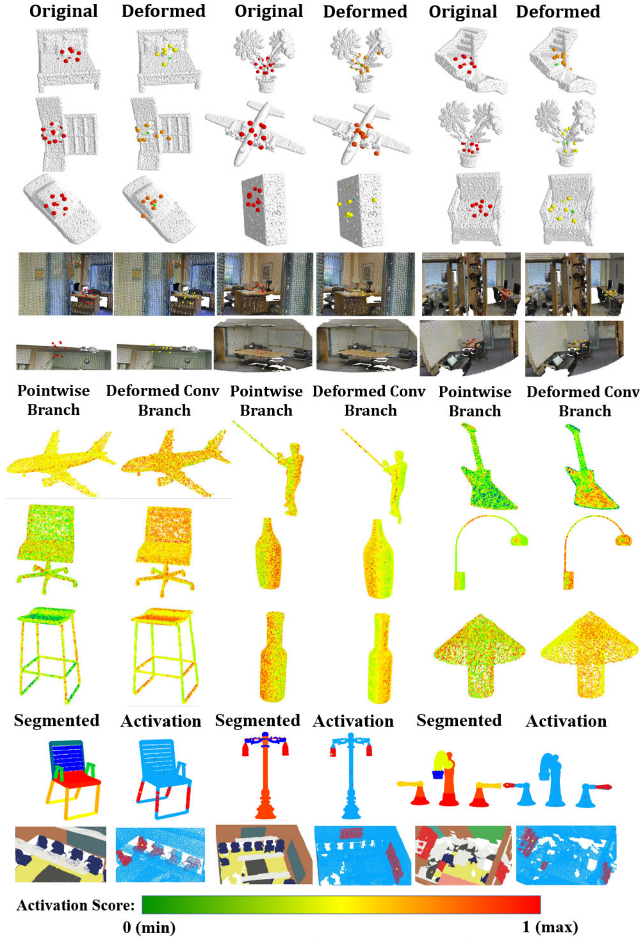


Fig. 10. Visualization of the deformable convolutional kernel, complementary features captured by two core network modules, and nonlocal activation. For deformable convolutions, left shows the original kernels, right shows the deformed kernels. The activation scores of pointwise correlation mining are shown on the left while the activation scores of deformable convolution based modeling are on the right. For nonlocal activations, left shows the segmentation predictions and right shows the nonlocal activations. (The background is indicated in blue and the query point is indicated in yellow, the red points are given large attentional weights.) Zoom in for details.

F. Ablation Study of Network Modules

Our designed network modules can be easily integrated seamlessly to existing point cloud backbones. Ablation studies

TABLE V
SEGMENTATION PERFORMANCE OF ABLATED NETWORK ON S3DIS

Ablation	mIOUs (%)
Remove pointwise feature relation mining (PFM)	66.2
Remove geometric convolutional modelling (GCM)	59.8
Remove attentional aggregation (AG)	67.1
Remove global feature extraction and AG	63.5
RandLA Remove attention Module	61.3
The full network framework	**70.8**
Without semantic context loss $L_2(\boldsymbol{W})$	68.2
With 2 RLB2 in each convolutional block	69.7
Choose $M = 1$ in RLB1 and RLB2	70.6

are also done to validate the effectiveness and necessity of our designed modules. As shown in Table V, core modules are removed from our network, respectively, and the mIOUs of 6-fold cross-validation on S3DIS benchmark is recorded. From the results, removing GCM results in 11% performance drop because learning the intrinsic geometric shape contexts of point clouds is vital for the recognition. On the other hand, removing global and local correlated feature mining results in 5.6% and 7.3% drop in mIOUs, which demonstrates both the local and long-range feature relationship capturing are also essential to the segmentation task. Not using the AG will also decline the performance for not retaining some meaningful features. Furthermore, the 5-stage ($h = 5$) network is the best choice because shallow networks have a poor fitting ability, while deeper networks will result in oversampling of point clouds, which will all deteriorate the performance. We also tested $M = 1$ in the RLB, however the segmentation performance does not increase. Therefore, $M = 8$ is used for memory efficiency.

V. Conclusion

In this work, we have proposed a general solution *FG-Net* to large-scale point cloud understanding with real-time speed and SOTA performance. The filtering and sampling methods are specially designed to improve the efficiency of large-scale point cloud processing, and they will boost the scene parsing performance. The designed network can effectively model point cloud structures and find the feature correlations across a long spatial range. Leveraging feature pyramid-based residual learning, hierarchical features at different resolutions can be fused in a memory efficient way. Extensive experiments on challenging real-world complex circumstances demonstrated that our approach outperforms SOTA methods in terms of performance, speed, and memory efficiency.

References

[1] F. Wang, Y. Zhuang, H. Zhang, and H. Gu, “Real-time 3-D semantic scene parsing with LiDAR sensors,” *IEEE Trans. Cybern.*, vol. 52, no. 3, pp. 1351–1363, Mar. 2022.

[2] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, “3D object recognition in cluttered scenes with local surface features: A survey,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2270–2287, Nov. 2014.

[3] Y. Cong, D. Tian, Y. Feng, B. Fan, and H. Yu, “Speedup 3-D texture-less object recognition against self-occlusion for intelligent manufacturing,” *IEEE Trans. Cybern.*, vol. 49, no. 11, pp. 3887–3897, Nov. 2019.

[4] A. Boulch, G. Puy, and R. Marlet, “FKAConv: Feature-kernel alignment for point cloud convolution,” in *Proc. 15th Asian Conf. Comput. Vis. (ACCV)*, 2020, pp. 1–26.

[5] I. Armeni *et al.*, “3D semantic parsing of large-scale indoor spaces,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 1534–1543.

[6] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, “Aligning point cloud views using persistent feature histograms,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems (IROS)*, 2008, pp. 3384–3391.

[7] R. B. Rusu, N. Blodow, and M. Beetz, “Fast point feature histograms (FPFH) for 3D registration,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2009, pp. 3212–3217.

[8] F. Tombari, S. Salti, and L. D. Stefano, “Unique signatures of histograms for local surface description,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 356–369.

[9] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “PointNet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Assoc., 2017, pp. 5099–5108.

[10] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, “Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 5589–5598.

[11] B. Graham, M. Engelcke, and L. van der Maaten, “3D semantic segmentation with submanifold sparse convolutional networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 9224–9232.

[12] C. Choy, J. Gwak, and S. Savarese, “4D spatio-temporal convnets: Minkowski convolutional neural networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 3075–3084.

[13] L. Landrieu and M. Boussaha, “Point cloud oversegmentation with graph-structured deep metric learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 7440–7449.

[14] X. Yang, X. Gao, B. Song, and B. Han, “Hierarchical deep embedding for aurora image retrieval,” *IEEE Trans. Cybern.*, vol. 51, no. 12, pp. 5773–5785, Dec. 2021.

[15] K. Yan, X. Wang, J. Kim, and D. Feng, “A new aggregation of DNN sparse and dense labeling for saliency detection,” *IEEE Trans. Cybern.*, vol. 51, no. 12, pp. 5907–5920, Dec. 2021.

[16] Y. Chen and X. Lu, “Deep category-level and regularized hashing with global semantic similarity learning,” *IEEE Trans. Cybern.*, vol. 51, no. 12, pp. 6240–6252, Dec. 2021.

[17] K. Liu, X. Han, and B. M. Chen, “Deep learning based automatic crack detection and segmentation for unmanned aerial vehicle inspections,” in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, 2019, pp. 381–387.

[18] K. Liu, Z. Gao, F. Lin, and B. M. Chen, “FG-Conv: Large-scale LiDAR point clouds understanding leveraging feature correlation mining and geometric-aware modeling,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2021, pp. 12896–12902.

[19] K. Liu, Y. Zhao, Z. Gao, and B. M. Chen, “Weaklabel3D-net: A complete framework for real-scene LiDAR point clouds weakly supervised multitasks understanding,” in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2022.

[20] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, “Rangenet++: Fast and accurate LiDAR semantic segmentation,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2019, pp. 4213–4220.

[21] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, “SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud,” in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2019, pp. 4376–4382.

[22] A. Kundu *et al.*, “Virtual multi-view fusion for 3D semantic segmentation,” 2020, *arXiv:2007.13138*.

[23] L. Li, S. Zhu, H. Fu, P. Tan, and C.-L. Tai, “End-to-end learning local multi-view descriptors for 3D point clouds,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 1919–1928.

[24] 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. (CVPR)*, 2017, pp. 652–660.

[25] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, “PointCNN: Convolution on X-transformed points,” in *Advances in Neural Information Processing Systems*. Red Hook, NY, USA: Curran Assoc., 2018, pp. 820–830.

[26] 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. (CVPR)*, 2020, pp. 12956–12964.

[27] 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. (CVPR)*, 2018, pp. 4548–4557.

[28] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 6411–6420.

[29] 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.

[30] J. Mao, X. Wang, and H. Li, "Interpolated convolutional networks for 3D point cloud understanding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1578–1587.

[31] G. Wang, Y. Yang, H. Zhang, Z. Liu, and H. Wang, "Spherical interpolated convolutional network with distance-feature density for 3D semantic segmentation of point clouds," 2020, *arXiv:2011.13784*.

[32] G. Wang, M. Chen, H. Liu, Y. Yang, Z. Liu, and H. Wang, "Anchor-based spatio-temporal attention 3D convolutional networks for dynamic 3D point cloud sequences," 2020, *arXiv:2012.10860*.

[33] W. Wu, Z. Qi, and L. Fuxin, "PointConv: Deep convolutional networks on 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 9621–9630.

[34] Z. Liu, H. Tang, Y. Lin, and S. Han, "Point-voxel CNN for efficient 3D deep learning," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2019, pp. 965–975.

[35] Q. Hu *et al.*, "Learning semantic segmentation of large-scale point clouds with random sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, May 25, 2021, doi: 10.1109/TPAMI.2021.3083288.

[36] S. Shi *et al.*, "PV-RCNN: Point-voxel feature set abstraction for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 10529–10538.

[37] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable ConvNets V2: More deformable, better results," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 9308–9316.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1–12.

[39] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, and C. Pan, "Densepoint: Learning densely contextual representation for efficient point cloud processing," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 5239–5248.

[40] G. Li, M. Muller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNS go as deep as CNNs?" in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 9267–9276.

[41] L. Liu *et al.*, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, 2020.

[42] J. Wang *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2021.

[43] J. Behley, V. Steinhage, and A. B. Cremers, "Efficient radius neighbor search in three-dimensional point clouds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2015, pp. 3625–3630.

[44] Z. Zhang, B.-S. Hua, and S.-K. Yeung, "ShellNet: Efficient point cloud convolutional neural networks using concentric shells statistics," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 1607–1616.

[45] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[46] B. Stadie *et al.*, "The importance of sampling inmeta-reinforcement learning," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2018, pp. 9280–9290.

[47] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–20.

[48] I. Lang, A. Manor, and S. Avidan, "SampleNet: Differentiable point cloud sampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 7578–7588.

[49] X. Roynard, J.-E. Deschaud, and F. Goulette, "Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification," *Int. J. Robot. Res.*, vol. 37, no. 6, pp. 545–557, 2018.

[50] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 9297–9307.

[51] M. Xu, R. Ding, H. Zhao, and X. Qi, "PAConv: Position adaptive convolution with dynamic kernel assembling on point clouds," 2021, *arXiv:2103.14635*.

[52] Z. Wu *et al.*, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 1912–1920.

[53] 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.

[54] K. Mo, S. Zhu, A. X. Chang, L. Yi, S. Tripathi, L. J. Guibas, and H. Su, "PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 909–918.

[55] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.Net: A new large-scale point cloud classification benchmark," in *Proc. Int. Soc. Photogram. Remote Sens.*, 2017, pp. 91–98.

[56] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 5828–5839.

[57] Z. Gao, G. Zhai, J. Yan, and X. Yang, "Pai-Conv: Permutable anisotropic convolutional networks for learning on point clouds," 2020, *arXiv:2005.13135*.

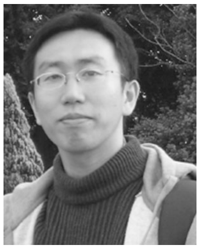
Kangcheng Liu (Student Member, IEEE) received the B.Eng. degree in electrical engineering and automation from the Harbin Institute of Technology, Harbin, China, in 2018. He is currently pursuing the Ph.D. degree with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Hong Kong.

His research interests include power systems analysis and control, robotics, LIDAR-SLAM, machine learning, and computer graphics for unmanned autonomous systems. Currently, he has strong interests in 3-D deep learning and scene understanding for robotic perception.

Zhi Gao received the B.Eng. and Ph.D. degrees from Wuhan University, Wuhan, China, in 2002 and 2007 respectively.

He has been with the Interactive and Digital Media Institute, National University of Singapore (NUS), Singapore, as a Research Fellow (A) and a Project Manager since 2008. In 2014, he joined the Temasek Laboratories, NUS, as a Research Scientist (A) and a Principal Investigator. He is currently working as a Full Professor with the School of Remote Sensing and Information Engineering, Wuhan University. Since 2019, he has been supported by the Distinguished Professor Program of Hubei Province and the National Young Talent Program, China. He has published more than 70 research papers on top journals and conferences, such as the *International Journal of Computer Vision*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *ISPRS Journal of Photogrammetry and Remote Sensing*, *Neurocomputing*, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, CVPR, ECCV, ACCV, and BMVC. His research interests include computer vision, machine learning, remote sensing, and their applications. In particular, he has strong interests in vision for intelligent systems and intelligent system-based vision.

Prof. Gao serves as an Associate Editor for the *Unmanned Systems*.



Feng Lin received the B.Eng. degree in computer science and control, and the M.Eng. degree in system engineering from Beihang University, Beijing, China, in 2000 and 2003, respectively, and the Ph.D. degree in computer and electrical engineering from the National University of Singapore, Singapore, in 2011.

He has been working as an Associate Research Scientist with the Peng Cheng Laboratory, Shenzhen, China, since 2019. His main research interests are unmanned aerial vehicles, vision-aided control and navigation, as well as embedded vision systems.

Dr. Lin was the recipient of the Best Application Paper Award, 8th World Congress on Intelligent Control and Automation, Jinan, China, in 2010. He has served for the editorial board of *Unmanned Systems*.

Ben M. Chen (Fellow, IEEE) received the B.S. degree in mathematics and computer science from Xiamen University, Xiamen, China, in 1983, the M.S. degree in electrical engineering from Gonzaga University, Spokane, WA, USA, in 1988, and the Ph.D. degree in electrical and computer engineering from Washington State University, Pullman, WA, USA, in 1991.

He has been a Professor of Mechanical and Automation Engineering with the Chinese University of Hong Kong, Hong Kong, since 2018. He was a Provost's Chair Professor with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore, where he also served as the Director of Control, Intelligent Systems and Robotics Area. He has authored/coauthored about 500 journal and conference articles, and ten research monographs in control theory and applications, unmanned systems, and financial market modeling. His current research interests are in unmanned systems and control applications.

Prof. Chen has received a number of research awards. His research team has actively participated in international UAV competitions and won many championships in the contests. He had served on the editorial boards of a dozen international journals, including *Automatica* and IEEE TRANSACTIONS ON AUTOMATIC CONTROL. He currently serves as an Editor-in-Chief for *Unmanned Systems* and a Deputy Editor-in-Chief for *Control Theory and Technology*. He is a Fellow of the Academy of Engineering, Singapore.