PointGS: Bridging and Fusing Geometric and Semantic Space for 3D Point Cloud Analysis

Chenru Jiang[a], Kaizhu Huang[b,*], Junwei Wu[a], Xinheng Wang[b], Jimin Xiao[c], Amir Hussain[d]

[a]*Department of Computer Science, University of Liverpool, Liverpool L69 7ZX, U.K.*
[b]*Data Science Research Center, Duke Kunshan University, No. 8 Duke Avenue, Kunshan, 215316, China.*
[c]*Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China.*
[d]*School of Computing, Edinburgh Napier University, Edinburgh, EH11 4BN, UK.*

Abstract

Directly processing 3D point cloud data becomes dominant in classification and segmentation tasks. Present mainstream point based methods usually focus on learning in either geometric space (*i.e.* PointNet++) or semantic space (*i.e.* DGCNN). Owing to the irregular and unordered data property of point cloud, these methods still suffer from drawbacks of either ambiguous local features aggregation in geometric space or poor global features extraction in semantic space. While few prior works address these two defects simultaneously by fusing information from the dual spaces, we make a first attempt to develop a synergistic framework, called PointGS. Leveraging both the strength of geometric structure and semantic representation, PointGS establishes a mutual supervision mechanism that can bridge the two spaces and fuse complementary information for better analyzing 3D point cloud data. Compared with existing popular networks, our work attains obvious performance improvement on all three mainstream tasks without any sophisticated operations. The code is publicly available at *https://github.com/ssr0512/PointGS*

*Corresponding author
Email addresses: chenru.jiang@liverpool.ac.uk (Chenru Jiang), kaizhu.huang@dukekunshan.edu.cn (Kaizhu Huang), junwei.wu@liverpool.ac.uk (Junwei Wu), xinheng.wang@xjtlu.edu.cn (Xinheng Wang), jimin.xiao@xjtlu.edu.cn (Jimin Xiao), A.Hussain@napier.ac.uk (Amir Hussain)

Keywords: Point Cloud, Geometric Space Learning, Semantic Space Learning, Information Fusion

1. Introduction

With recent advances in 3D scanning technologies, it becomes convenient to obtain 3D raw data. As the fundamental 3D representation, point cloud has attracted extensive attention for various 3D applications [1, 2]. Recently, researchers focus on exploiting Convolution Neural Networks (CNNs) to process 3D point cloud, which can be generally categorized into three types: projection-based methods [3, 4, 5], voxelization-based methods [6, 7], and point-based methods [8, 9, 10, 11]. Among them, point-based methods, processing point sets directly with Multi-Layer Perceptrons (MLP), have become dominant due to their efficiency and high performance.

Point-based approaches adopt raw point cloud data as inputs which can be further classified into two learning strategies. The first strategy focuses on features aggregation in the geometric space like PointNet [8], PointNet++ [9], and other extensions [12, 13, 14, 15, 16]. Recently, PointNeXt [17] focuses on training skills and scale strategies to improve Pointnet++ performance further; PointMLP [18] introduces a residual MLP structure to re-explore the CNN capability without any sophisticated local feature extractor design; ASSANET [19] redesigns the set abstraction module in PointNet++ for reducing the anisotropy of neighbor features. All these methods iteratively apply Farthest Point Sampling (FPS) algorithms to downsample original 3D point cloud. For information aggregation, kNN is adopted on downsampled data to search neighbors of each point in the geometric space. As illustrated in Figure 1 (a), point cloud gradually becomes sparser after each FPS while kNN (with a fixed k) could obtain a bigger search scope to aggregate more non-local points so that rich global information is extracted from different scale datasets. Yet, one inherited defect of this strategy is that kNN search may fail to aggregate the same-category points in the geometric space, e.g. the four blue points on the different chair legs shar-

ing the same classification category, as illustrated in Figure 1 (a). In fact, the Euclidean distances (blue lines) among them are still much bigger than multiple kNN search radius (radiuses of 3 blue circles). Consequently, these four points are less possible to be aggregated by kNN, thus limiting the model accuracy of learning in the geometric space.

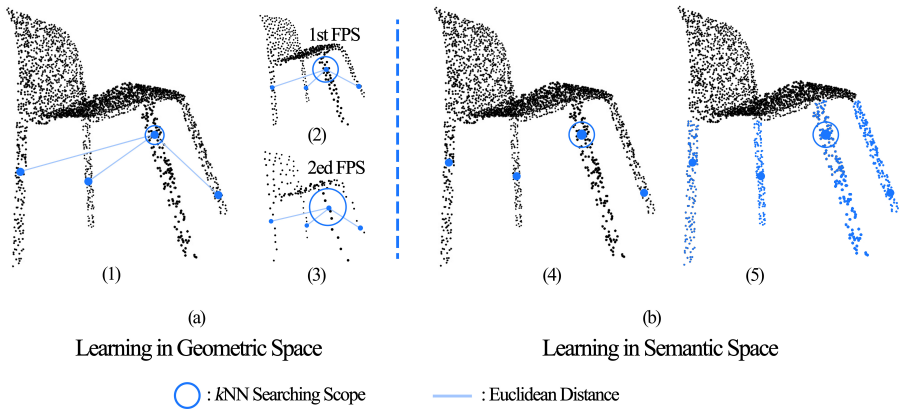

Figure 1: Illustration of learning in geometric space or in semantic space. In (a), kNN obtains a big search scope on sparse dataset, where the four blue points of chair legs however can hardly be aggregated due to the limited search radius. In (b), the same-category features can be readily aggregated together in semantic space. Yet, kNN unfortunately may not get global features efficiently due to non-downsampled data.

The second strategy focuses on aggregating information or features in the semantic space, such as DGCNN [10] and others [20, 21, 22, 23]. AdaptiveGraph [24] proposes to assign learning weights on each edge for better information evaluation and aggregation; Grid-GCN [25] develops a novel grid context aggregation strategy on edge features in order to attain a good balance between efficiency and performance; 3D-GCN [26] learns unique 3D kernels with the graph max-pooling mechanism for robust feature aggregation. This kind of approaches encode the original 3D coordinate information into semantic representations in the first layer, and then consistently handle the feature learning in semantic space without geometric information supplement. Since all points have been transformed into the high-level features which contain robust representation, the same-category point aggregation becomes more straightforward.

As illustrated in Figure 1 (b), all the points of chair legs, which have similar features, are readily aggregated together after multiple kNN search process in the semantic space. However, geometric structures of objects are unavailable in semantic space. As such, it might be improper to apply FPS to downsample semantic features. Thus, kNN is unfortunately inefficient to extract global information on the non-downsampled semantic features. Though graph pooling or increasing the search radius can be considered as possible solutions in the semantic space, the computational cost is prohibitive. On the other hand, the redundant local features updating process also hinders the model efficiency.

In this paper, we utilize the strength of geometric space and semantic space to develop a novel dual-space information fusion mechanism for simultaneously addressing the limitations of both learning strategies. For global features extraction, we successively apply FPS on original point cloud to generate multi-scale geometric information as shown in Figure 2 (a). To enrich the geometric priors, we calculate spherical coordinates relation which is separately combined with different scale point cloud to form inputs. Then, we separately input these data into different branches of PointGS for delicate feature extraction where the rich global features can be obtained. In particular, FPS indexes are preserved end-to-end which contain robust geometric prior knowledge of objects. We then utilize these fixed indexes to downsample features straightforwardly in semantic space for eschewing high computational graph pooling process or large search radius setting. Detailed justification can be seen in Section 3. For same-category point aggregation within each branch of PointGS, we iteratively utilize kNN in semantic space in order to avoid the dilemma as demonstrated in Figure 1 (a). Meanwhile, we explore a local feature updating rule to simplify the aggregation process. In addition, we conduct a fusion mechanism to reinforce features communication among different branch of PointGS as illustrated in Figure 2 (b). Specifically, our framework presents an elegant information fusion system that acts as a bridge between Geometric and Semantic spaces to alternatively reinforce feature fusion and interaction between two spaces while attaining a mutual supervision mechanism. As demonstrated in Figure 2 (c), 3D points

are first transformed into semantic features, and kNN indexes of semantic space are utilized to find corresponding points in geometric space for more relevant geometric information supplement. Then, the selected points are encoded and fed back into semantic space to further calibrate semantic feature aggregation. Such fusion process is iteratively applied between two spaces so that PointGS not only ensures original data structure consistency in both spaces, but also forms a dual-space mutually supervised learning mechanism.

Without sophisticated operations, PointGS exhibits superior performance on 3D point cloud analysis and achieve comparable results with state-of-the-art methods on classification and segmentation tasks. The major contributions of this paper are summarized as follows.

- To efficiently capture global information, our model utilizes FPS to form multi-scale geometric inputs which are helpful to extract rich global information and eschew complex pooling in semantic space.
- To effectively aggregate similar local features/information, we iteratively apply kNN in semantic space for similar features aggregation and explore a new feature updating rule for simplifying the aggregation process.
- To simultaneously address the defects of two learning strategies, we design a dual-space information fusion architecture, in order to establish a mutual supervision mechanism between geometric and semantic spaces.

2. Related Work

2.1. Voxelization-based and Projection-based Learning

Voxelization-based and Projection-based methods transform the point cloud into an ordered data form in order to take advantages of powerful CNNs. Many voxelization-based works [6, 27, 28, 29] map the points into the regular 3D grid representations and apply 3D convolutions. However, the massive computation costs posit challenges in these approaches due to the cubic growth in the number of voxels. To improve efficiency, OctNet [7] and Kd-Net [30] utilize tree models

and skip the empty voxels. Albeit their efficiency, these strategies still suffer from information loss during the quantification process on the voxel grid. Alternatively, the projection-based methods [31, 4, 32, 33] project 3D points to a set of multi-view 2D images, make it possible to apply traditional 2D convolutions directly. Nevertheless, 2D CNN operations used in these methods make it less possible to capture non-local geometric features. Moreover, they often struggle with points inner occlusion. In this paper, we follow the point-based learning strategy but explores a simpler yet effective architecture.

2.2. Point-based Learning in Geometric Space

3D coordinates constitute point cloud data in geometric space. PointNet [8] is the pioneering work to take these raw data as inputs which demonstrates the possibility of processing irregular point clouds directly. For better locality encoding, PointNet++ [9] further applies PointNet as set abstraction mechanism and a hierarchical structure to aggregate local features. After PointNet++, numerous works focus is shifted to how to generate better regional point representations. PointConv [34] and KPConv [12] focus on constructing convolution weights matrices based on the input coordinates. PointCNN [15] permutes the neighbor points to a fixed order, thus enabling to apply convolutions directly. InterpCNN [14] utilizes coordinates to interpolate pointwise kernel weights, and SpiderCNN [35] defines kernel weights as a family of polynomial functions. Recently, ASSANET [19] redesigns the set abstraction module in PointNet++ for reducing the anisotropy of neighbor features. PointNeXt [17] focuses on training skills and scale strategies to improve Pointnet++ performance further. PointMLP [18] introduces a residual MLP structure to re-explore the CNN capability without any sophisticated local feature extractor design. All these methods focus on iteratively utilizing spatial coordinates to learn 3D structure relations. However, as illustrated in Figure 1 (a), kNN search in geometric space might be limited to aggregate same-category points. In this paper, we showcase that even without applying the carefully designed convolution process of local set abstraction, a small modification of learning strategy is able to exhibit

gratifying performance and even better results.

2.3. Point-based Learning in Semantic Space

This kind of approaches encode 3D coordinates at first and then focus on conducting message passing in semantic space. After coordinates encoding in the first layer, DGCNN [10] proposes the EdgeConv model to aggregate edge representations of point cloud and search similar features in semantic space. ECC [20] aims to use edge information to generate conditional edge filters. AdaptiveGraph [24] proposes to assign learning weights on each edge for better information evaluation and aggregation. Grid-GCN [25] develops a novel grid context aggregation strategy on edge features in order to attain a good balance between efficiency and performance. 3D-GCN [26] learns unique 3D kernels with the graph max-pooling mechanism for robust feature aggregation. GACNet [36] employs graph attention convolution, and SPG [37] operates on a superpoint graph to represent contextual relations. However, the low efficiency global feature extraction and edge information redundancy usually hinder the performance of this kind of method. PCT [38] and Point Transformer [39] utilize transformer structure to capture long-range relation within point cloud, and their complicated structure commonly incur unfavorable computational cost. Compared with these transformer-based methods, PointGS is more simple and exhibits competitive or even better performance. In this paper, we attain convenient global information extraction without sophisticated operators and explore an efficient local features updating rule for point cloud understanding.

3. Main Methodology

3.1. Geometric Space Learning

Global Information Extraction and Guidance Within our method, we first utilize FPS on the original geometric data to generate multi-scale inputs and reserve FPS indexes as a geometric guidance for the subsequent downsampling process. Different from conventional geometric space learning approaches

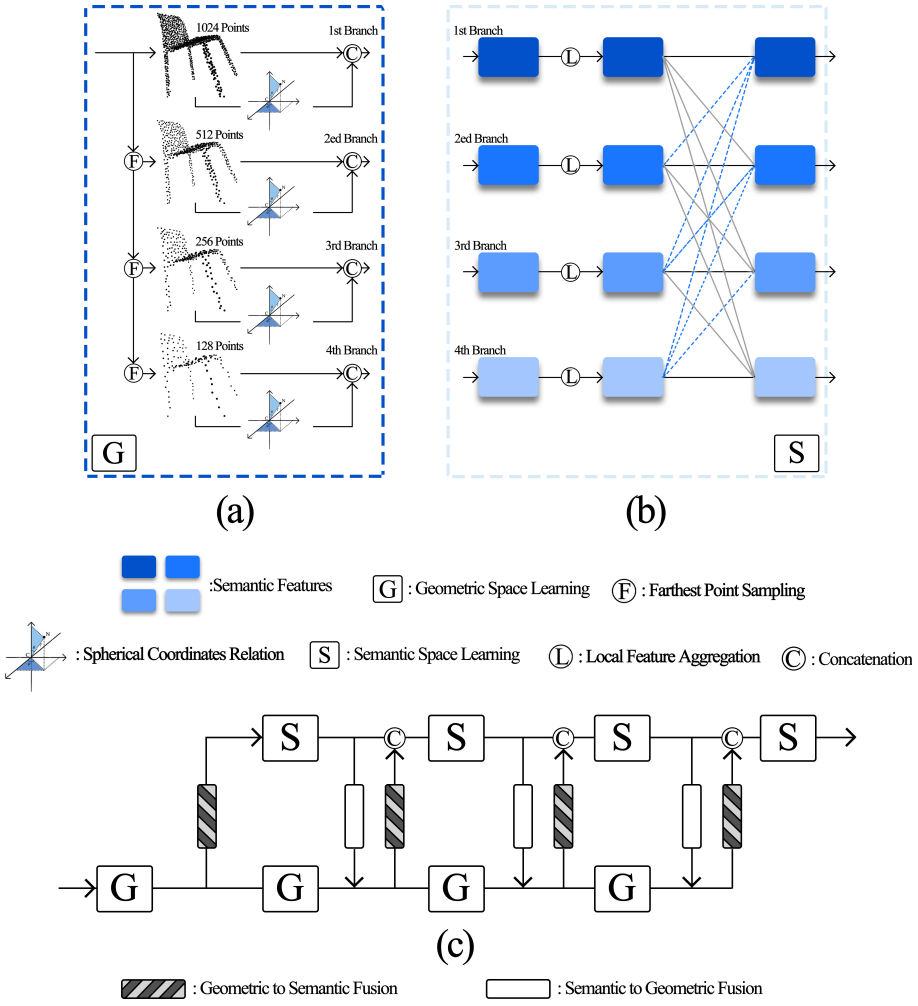

Figure 2: (a) shows the geometric space learning structure in PointGS, where FPS downsampling is successively utilized to generate multi-scale inputs. The spherical coordinate relation of each scale point cloud are concatenated together with corresponding inputs as extra initial information supplement. (b) illustrates semantic space learning details of PointGS. We conduct a simple graph model to aggregate semantic features and present a fusion mechanism to to enhance information communication among four branches. (c) details the dual-space fusion structure, enabling to achieve alternatively learning in either geometric or semantic space.

like PointNet++ [9], which apply FPS within the different model stages, we successively apply FPS at the beginning to form a pyramid geometric information as shown in Figure 2 (a). After concatenating with corresponding spherical

coordinate relation, each scale inputs are simultaneously encoded into semantic space by different branches of PointGS. Benefiting from pyramid structure, global information extraction becomes efficient and convenient. Intuitively, the 4-scale feature learning process is independent which hinders the feature communication among different scale features. Feature fusion among multiple branches is one straightforward solution to handle this problem, yet downsampling in semantic space is still a challenge. To this end, we propose to directly leverage FPS indexes as guidance, which termed as Geometric Information Guidance (GIG), to efficiently attain semantic features downsampling within feature fusion process. The FPS indexes contain consistent and robust object geometric structure knowledge which ensures the rationality of the downsampling process in semantic space. For upsampling, the simple feature interpolation of Pointnet++ is employed in our method to attain reliable and efficient feature upsampling. The right part of Figure 2 (b) details the structure of proposed feature fusion process. Grey lines indicate the dowsampling process which adopt FPS indexes as guidance, and blue dash lines represent the simple upsampling process.

Albeit its simplicity, the proposed structure exhibits some prominent merits. 1) By constructing multi-scale inputs and multi-branch learning structure, global information extraction becomes efficient and convenient. 2) Once FPS is applied at the beginning to form multi-scale inputs, time-consuming downsampling is unnecessary in the semantic space anymore. Meanwhile, the FPS indexes can be efficiently utilized to downsample semantic representations within multiple feature fusion process. 3) Constant FPS indexes from end-to-end ensure the geometric structure consistency at different scale data which provides a geometric guidance for reasonable downsampling in semantic space. The effectiveness of proposed downsampling mechanism can be clearly seen in Table 1, and more details can be later seen in Section 4.4.

Geometric Relation Supplement Spatial geometric information is critical for model performance [40]. In order to provide more geometric information, we transform coordinates at each scale and their neighbours into the spherical coordinate system to obtain relative angles for more spatial relation description

Table 1: Comparison of with/without Geometric Information Guidance (GIG).

	ModelNet40				ShapeNetPart			
Model	Input	mAcc	OA		Input	Cls. mIou	Ins. mIou	
Without GIG	1k	90.1	93.0		2k	82.0	85.8	
With GIG	1k	**90.9**	**93.8**	0.8 ↑	2k	**82.8**	**86.6**	0.8 ↑

among points. Figure 3 details the relative angle expression based on three axes separately. N is one neighbour point and C is the center point. By adopting *arctan* function, the polar angle (θ_i) and the azimuth angle (φ_i) are calculated through two points coordinates. γ_i is the radial distance between neighbor and center point ($i = X, Y, Z$), which is the same as Euclidean distance and we dropped in PointGS for simplicity. According to different axes, we obtain three pairs of relevant angle relation (θ_X, φ_X, θ_Y, φ_Y, θ_Z, φ_Z) as extra spatial geometric information for the model. By doing so, we enable PointGS to obtain rich geometric information while maintaining the simplicity of the model structure. The effectiveness of these geometric relation supplement is later empirically verified in Section 4.4.

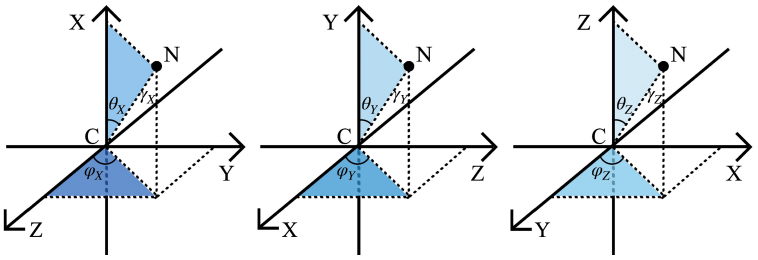

Figure 3: Spherical coordinate system illustration in this work. θ_i is the polar angle, φ_i is the azimuth angle, and γ_i is the radial distance between N and C ($i = X, Y, Z$). Polar angles and azimuth angles are adopted as extra relevant relations information.

3.2. Semantic Space Learning

Revisiting Conventional Local Features Updating Rule For reliable local features aggregation, we focus on feature learning in semantic space and modify the local features updating rule for efficient purpose. By convention, the local features updating rule of GNNs on point cloud, like DGCNN, can be formulated as:

$$f_i^{l+1} = \Lambda_{j \in N(i)} \Phi(f_i^l || f_j^l - f_i^l). \tag{1}$$

$\Lambda(\cdot)$ means the aggregation function (max-pooling), $\Phi(\cdot)$ denotes the local features extraction function (MLPs), and f_i^l represents the semantic features of point i at layer l. $N(i)$ is the set of neighbor features, which is identified by kNN search (k number is 20-40) in semantic space. Different from traditional graph learning field, no adjacency matrix [41] is available to indicate topological information in point cloud. Thus, the edge information is commonly redundant. According to [40], it is feasible to simplify the local features updating rule with negligible performance drop. They present a simple version to distill edges:

$$f_i^{l+1} = \Lambda_{j \in N(i)} \Phi_1(f_j^l) - \Phi_2(f_i^l). \tag{2}$$

Equation (2) passes neighbor point features (f_j^l) through MLPs (Φ_1) and directly applies the aggregation function $\Lambda(\cdot)$ to obtain only one neighbor. Then, the resulting features (f_i^{l+1}) are subtracted from the distilled neighbor features. Such rule can nearly achieve 20-40 (k number) times reduction in computational cost. In a similar way, we modify the local features updating rules of PointGS to compact semantic features aggregation process.

Local Features Updating on Classification and Part Segmentation In our work, the information within the first local features updating is not simplified since geometric priors can serve as a cheap and effective way to boost model accuracy [8, 42]. Thus, the first local features updating function can be written as follows:

$$f_i^{l+1} = \Lambda_{j \in N(i)} \Phi(p_j^l - p_i^l || p_j^l || r_{\theta_X}^l || r_{\varphi_X}^l || r_{\theta_Y}^l || r_{\varphi_Y}^l || r_{\theta_Z}^l || r_{\varphi_Z}^l || dist^l). \tag{3}$$

The first layer inputs are spatial coordinates p, f means the semantic features after MLPs (Φ), and $dist^l$ is Euclidean distance between neighbors and center. In order to provide more initial information, we not only use edge information $(p_j^l - p_i^l)$, but also employ spherical coordinate angle relation $(r_{\theta_X}^l, r_{\varphi_X}^l$ ...) as geometric priors supplement. Here we do not use the coordinates of center points for efficiency purpose, which can be directly computed from the neighbor coordinates (p_j^l) and edges. From the second local feature aggregation, we start to distill features while repeatedly injecting original spatial relation to supply geometric information. The proposed updating rule is as follows:

$$\begin{aligned} f_i^{l+1} &= \Lambda_{j \in N(i)} \Phi_1(f_j^l || f_{supply}^l) - \Phi_2(f_i^l), \\ f_{supply}^l &= \Phi_3(p_j^l - p_i^l || p_j^l || r_{\theta_X}^l || r_{\varphi_X}^l || r_{\theta_Y}^l || r_{\varphi_Y}^l || r_{\theta_Z}^l || r_{\varphi_Z}^l). \end{aligned} \tag{4}$$

Different form Equation (2), we propose to inject extra spatial relation within each features updating process as the geometric information supplement that the supplemented features form (f_{supply}) is similar with Equation 3 but without aggregation function and Euclidean distance. Then, the concatenation results of supplemented features and neighbor features (f_j) are directly processed by aggregation function in order to achieve redundancy reduction as Equation (2). As demonstrated in Equation (4), the neighbor indexes (j) within f_{supply} is the same with f_j. Thus, the neighbor points selection in geometric space is based on the results of semantic space learning which belongs to our dual-space fusion mechanism and we detailed in Section 3.3. On the other hand, we notice that Euclidean distances ($dist^l$) would largely hamper the performance after the first local aggregation. The reason could be that the Euclidean distance in geometric space mismatches with features in semantic space. As analyzed in Section 1, the Euclidean distances of same-category points are big in geometric space (Figure 1 (a)) but the feature distances are small in semantic space. As a result, we discard this geometric space information after the first local features updating. Table 2 lists out the model performance with/without Euclidean distance after the first local feature aggregation on ModelNet40 and ShapeNetPart, where the results clearly support our analysis.

Table 2: Comparison of with/without Euclidean Distance (ED).

	ModelNet40				ShapeNetPart			
Model	Input	mAcc	OA		Input	Cls. mIou	Ins. mIou	
Without ED	1k	88.2	92.9		2k	82.4	86.1	
With ED	1k	**90.9**	**93.8**	0.9 ↑	2k	**82.8**	**86.6**	0.5 ↑

Local Feature Aggregation on Semantic Segmentation Unlike the first two tasks, where the input data exclusively consists of 3D coordinates, semantic segmentation benchmark combines RGB color values as extra inputs. Built upon our analysis in Section 3.2, we present the following rule for this task in the first local feature aggregation:

$$f_i^{l+1} = \Lambda_{j \in N(i)} \Phi(p_j^l - p_i^l || p_j^l || r_{\theta_X}^l || r_{\varphi_X}^l || r_{\theta_Y}^l || r_{\varphi_Y}^l || r_{\theta_Z}^l || r_{\varphi_Z}^l || dist^l || dist_{rgb}^l). \quad (5)$$

In Equation (5), there is only one modification that the Euclidean distance form of RGB values is introduced to the updating rule as an extra relevant relation. Since the physical significance is irrelevant between the 3D coordinates and RGB information, we consider these two kinds of representations separately. In contrast to conventional methods, we transform RGB values to the Euclidean distances form ($dist_{rgb}^l$) as a relevant relation for the first updating function. But the subsequent updating rules keep the same as Equation (4) for avoiding features mismatch problem between geometric and semantic space.

We present an example to explain the Euclidean distance form of RGB values in Figure 4 where similar parts or objects have the similar colors, and vice versa. Subfigures A and B are examples of similar parts containing closer RGB colors for which their Euclidean distances are also small (0.5751). In contrast, for the different object parts like A, C and B, C, the RGB Euclidean distances are bigger which are 0.6336, 0.7909 respectively. According to this observation, we believe the Euclidean distance form of RGB is helpful for our model to

Table 3: Comparison results of with/without RGB Euclidean Distance (RGB_ED) on S3DIS Area5 dataset.

	S3DIS			
Model	OA	mAcc	mIou	
Without RGB_ED	86.0	66.4	57.5	
With RGB_ED	**87.7**	**70.0**	**61.7**	4.2 ↑

distinguish detailed structure information in more complex scenes. Table 3 lists the comparison experiments in order to verify the superiority (4.2% gain) of adopting the Euclidean distance form of RGB values, and Section 4.4 provides more supportive experiments.

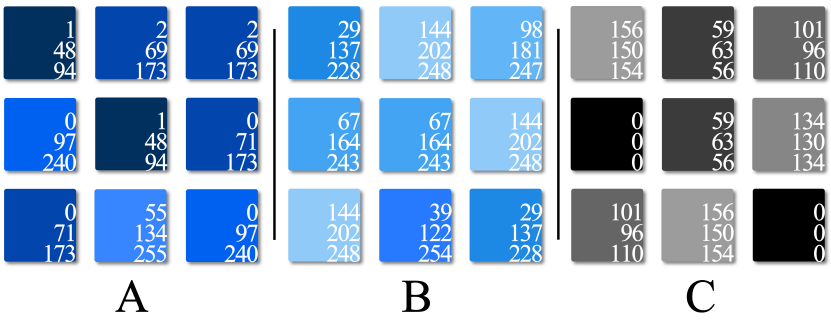

Figure 4: Illustration of the relevant relation among point cloud with RGB colors where similar points have closer RGB color and smaller Euclidean distance of RGB.

3.3. Dual-Space Information Fusion

To achieve information fusion learning between geometric space and semantic space, we explore a novel architecture as illustrated in Figure 2(c). In the process of Geometric to Semantic Fusion (GSF), we adopt MLP to transform original coordinates into semantic space for feature extraction. In semantic space, kNN is adopted on each point features to filter out similar neighbours for local feature aggregation. After that, we start Semantic to Geometric Fusion

(SGF) to select corresponding points in geometric space as the information supplement for the next GSF process. Particularly, kNN indexes of semantic space are utilized to select corresponding points which avoids the spatial limitation of searching in geometric space. As demonstrated in Equation (4), the supplementary geometric information are encoded by GSF (Φ_3) and the transformed features (f^l_{supply}) are concatenated with semantic features (f^l_j) for the following local features updating. With such dual-space fusion structure, we showcase two advantages. 1) Geometric information is vital to boost model performance [8, 42] and is iteratively feeded into PointGS, making our model become more robust. 2) The same-category coordinates selection according kNN indexes of semantic space can detour Euclidean distance restriction and aggregate more long-range same-category points for geometric prior supplement.

4. Experiments

In this section, we evaluate PointGS on several benchmark tasks including shape classification, part segmentation, and semantic segmentation. We also conduct ablation studies to examine extensively the effectiveness of our PointGS. For fair comparison, we follow the same data processing and evaluation protocols as used in PointNet++ [9].

4.1. Shape Classification

Data and Metric We first evaluate PointGS on the ModelNet40 [29], which contains 9,843 objects for training and 2,468 objects for testing meshed CAD models belonging to 40 categories. We randomly sample 1,024 points from each CAD objects which is more difficult than uniformly sample. The mean accuracy within each category (mAcc) and the overall accuracy (OA) are adopted to evaluate model performance.

Performance Analysis The shape classification results are presented in Table 4. As observed, compared with PointNet++ and DGCNN, performance gains of our model are 3.1% and 1.6%. Compared with the other popular meth-

Table 4: Shape classification results.

			ModelNet40		ModelNet10	
Method	Input	#Points	mAcc	OA	mAcc	OA
ECC [20]	xyz	1k	83.2	87.4	90.0	90.8
PointNet [8]	xyz	1k	86.0	89.2	-	-
Kd-Net [30]	xyz	1k	86.3	90.6	92.8	93.3
PointNet++ [9]	xyz	1k	-	90.7	-	-
KCNet [43]	xyz	1k	-	91.0	-	94.4
PointNet++ [9]	xyz, normal	5k	-	91.9	-	-
3D-GCN [26]	xyz	1k	-	92.1	-	-
SpecGCN [23]	xyz	1k	-	91.8	-	-
Grid-GCN^2 [25]	xyz	1k	89.7	92.0	95.3	**95.8**
PointCNN [15]	xyz	1k	88.1	92.2	-	-
DGCNN [10]	xyz	1k	90.2	92.2	-	-
PointWeb [13]	xyz	1k	89.4	92.3	-	-
PCNN [44]	xyz	1k	-	92.3	-	94.9
SpiderCNN [35]	xyz, normal	5k	-	92.4	-	-
KPConv [12]	xyz	7k	-	92.9	-	-
ASSANET(L) [19]	xyz	7k	-	92.9	-	-
InterpCNN [14]	xyz	1k	-	93.0	-	-
DRNet [45]	xyz	1k	-	93.1	-	-
PointASNL [46]	xyz, normal	1k	-	93.2	-	95.7
PCT [38]	xyz	1k	-	93.2	-	-
SO-Net [47]	xyz, normal	5k	-	93.4	-	95.7
AdaptiveGraph [24]	xyz	1k	90.7	93.4	-	-
PointPG (Ours)	xyz	1k	**90.9**	**93.8**	**95.8**	95.7

ods, our simple network attains the best performance in both metrics with only 1k points on ModelNet10 and ModelNet40 datasets.

Table 5: Part segmentation results.

Method	Cls. mIou	Ins. mIou	airplane	bag	cap	car	chair	earphone	guitar	knife	lamp	laptop	motorbike	mug	pistol	rocket	skateboard	table
Kd-Net [30]	77.4	82.3	80.1	74.6	74.3	70.3	88.6	73.5	90.2	87.2	81.0	94.9	57.4	86.7	78.1	51.8	69.9	80.3
PointNet [8]	80.4	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
SO-Net [47]	80.8	84.6	81.9	83.5	84.8	78.1	90.8	72.2	90.1	83.6	82.3	95.2	69.3	94.2	80.0	51.6	72.1	82.6
PCCN [16]	81.8	85.1	82.4	80.1	85.5	79.5	90.8	73.2	91.3	86.0	85.0	95.7	73.2	94.8	83.3	51.0	75.0	81.8
PointNet++ [9]	81.9	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
DGCNN [10]	82.3	85.1	84.2	83.7	84.4	77.1	90.9	78.5	91.5	87.3	82.9	96.0	67.8	93.3	82.6	59.7	75.5	82.0
3D-GCN [26]	82.7	85.3	82.8	86.1	84.8	79.2	91.9	74.9	91.6	87.4	83.6	95.8	69.3	94.9	82.4	61.1	75.6	82.2
SpiderCNN [35]	82.4	85.3	83.5	81.0	87.2	77.5	90.7	76.8	91.1	87.3	83.3	95.8	70.2	93.5	82.7	59.7	75.8	82.8
PointConv [34]	82.8	85.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SGPN [48]	82.8	85.8	80.4	78.6	78.8	71.5	88.6	78.0	90.9	83.0	78.8	95.8	77.8	93.8	**87.4**	60.1	**92.3**	**89.4**
PointCNN [15]	84.6	86.1	84.1	**86.5**	86.0	80.8	90.6	79.7	92.3	88.4	85.3	96.1	77.2	95.2	84.2	64.2	80.0	83.0
PointASNL [46]	-	86.1	84.1	84.7	87.9	79.7	92.2	73.7	91.0	87.2	84.2	95.8	74.4	95.2	81.0	63.0	76.3	83.2
ASSANET(L) [19]	-	86.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
RS-CNN [49]	84.0	86.2	83.5	84.8	88.8	79.6	91.2	**81.1**	91.6	88.4	86.0	96.0	73.7	94.1	83.4	60.5	77.7	83.6
InterpCNN [14]	84.0	86.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
DRNet [45]	-	86.4	84.3	85.0	88.3	79.5	91.2	79.3	91.8	**89.0**	85.2	95.7	72.2	94.2	82.0	60.6	76.8	84.2
KPConv [12]	**85.1**	86.4	84.6	86.3	87.2	81.1	91.1	77.8	**92.6**	88.4	82.7	**96.2**	**78.1**	**95.8**	85.4	**69.0**	82.0	83.6
PCT [38]	-	86.4	85.0	82.4	**89.0**	**81.2**	**91.9**	71.5	91.3	88.1	**86.3**	95.8	64.6	**95.8**	83.6	62.2	77.6	83.7
AdaptiveGraph [24]	83.4	86.4	**84.8**	81.2	85.7	79.7	91.2	80.9	91.9	88.6	84.8	**96.2**	70.7	94.9	82.3	61.0	75.9	84.2
PointPG (Ours)	82.8	**86.6**	85.2	85.3	86.0	80.5	92.0	73.9	92.3	88.5	85.9	**96.2**	63.3	94.9	82.2	59.4	75.8	83.2

4.2. Object Part Segmentation

Data and Metric We conduct experiments on ShapeNetPart [50] dataset containing 16 shape categories, where 14,006 3D objects are used for training and 2,874 for testing. The part number for each category is between 2 and 6, and there are 50 different parts in total. We use the same input sample strategy for fairness. In this dataset, category mIou and instance mIou are reported.

Performance Analysis Table 5 lists the model performance on ShapePartNet. Intuitively, our method demonstrates clear superiority to the existing popular methods. Compared with PointNet++ and DGCNN, our network attains 0.9%, 0.5% gains on class mIou and 1.5% improvement on instance mIou. Note that we did not adopt loss-balancing during the training process, which is supposed to be able to boost the performance further.

4.3. Semantic Segmentation

Data and Metric We next test PointGS for 3D semantic segmentation on the challenging Stanford Large-Scale 3D Indoor Spaces (S3DIS) dataset [51]. S3DIS consists of 271 rooms in six areas. The points of scene are assigned a semantic label from 13 categories (e.g. table, floor, wall). Following the conventional evaluation protocol [52, 9], our model is evaluated in 2 modes: (1) Area 5 is withheld during training and is adopted for testing, (2) 6-fold cross-validation. Mean class-wise intersection over union (mIou), mean of class-wise accuracy (mAcc), and overall pointwise accuracy (OA) are used as metrics.

Table 6: Semantic segmentation results on the S3DIS dataset, evaluated on Area 5.

Method	OA	mAcc	mIou	ceiling	floor	wall	beam	colimn	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [8]	-	49.0	41.1	88.8	97.3	69.8	0.1	3.9	46.3	10.8	59.0	52.6	5.9	40.3	26.4	33.2
SegCloud [52]	-	57.4	48.9	90.1	96.1	69.9	0.0	18.4	38.4	23.1	70.4	75.9	40.9	58.4	13.0	41.6
TangentConv [53]	-	62.2	52.6	90.5	97.7	74.0	0.0	20.7	39.0	31.3	77.5	69.4	57.3	38.5	48.8	39.8
PointNet++ [9]	82.8	60.2	53.1	88.9	97.6	72.8	0.0	10.4	56.7	6.8	70.5	79.8	36.1	62.9	61.8	46.3
DGCNN [10]	85.0	61.6	54.5	93.1	98.0	81.2	0.0	7.8	61.6	34.1	71.2	76.8	19.6	58.2	63.8	43.8
PointCNN [15]	85.9	63.9	57.3	92.3	98.2	79.4	0.0	17.6	22.8	62.1	74.4	80.6	31.7	66.7	62.1	56.7
SPGraph [37]	86.4	66.5	58.0	89.4	96.9	78.1	0.0	42.8	48.9	61.6	84.7	75.4	69.8	52.6	2.1	52.2
PCCN [16]	-	67.0	58.3	92.3	96.2	75.9	0.3	6.0	69.5	63.5	66.9	65.6	47.3	68.9	59.1	46.2
PiontWeb [13]	87.0	66.6	60.3	92.0	98.5	79.4	0.0	21.1	59.7	34.8	76.3	88.3	46.9	69.3	64.9	52.5
PCT [38]	-	67.7	61.3	92.5	98.4	80.6	0.0	19.4	61.6	48.0	76.6	85.2	46.2	67.7	67.9	52.3
Ours	**87.7**	**69.9**	**61.7**	92.2	97.9	82.6	0.0	25.9	53.4	69.8	74.9	80.3	38.6	66.2	69.6	51.1

Table 7: Semantic segmentation on S3DIS, evaluated with 6-fold cross-validation.

Method	OA	mAcc	mIou	ceiling	floor	wall	beam	colimn	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [8]	78.5	66.2	47.6	88.0	88.7	69.3	42.4	23.1	47.5	51.6	42.0	54.1	38.2	9.6	29.4	35.2
RSNet [54]	-	66.5	56.5	92.5	92.8	78.6	32.8	34.4	51.6	68.1	60.1	59.7	50.2	16.4	44.9	52.0
PointNet++ [9]	84.1	70.4	60.1	93.3	91.7	76.1	33.2	27.6	57.4	59.0	63.5	70.5	41.3	55.7	57.3	54.6
SPGraph [37]	85.5	73.0	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	73.5	69.2	63.2	45.9	8.7	52.9
DGCNN [10]	86.4	71.7	62.3	94.0	94.0	81.7	37.9	35.6	61.1	59.2	67.1	68.4	30.5	55.7	59.1	55.5
A-CNN [55]	87.3	-	62.9	92.4	96.4	79.2	59.5	34.2	56.3	65.0	66.5	78.0	28.5	56.9	48.0	56.8
PointCNN [15]	**88.1**	75.6	65.4	94.8	97.3	75.8	63.3	51.7	58.4	57.2	71.6	69.1	39.1	61.2	52.2	58.6
Ours	87.7	**76.5**	**66.5**	93.1	94.7	82.4	35.3	47.4	63.0	73.8	67.3	72.9	52.8	61.0	62.0	58.5

Performance Analysis We present the comparison results on Area 5 (Table 6), and 6-fold cross-validation (Table 7). In this complex indoor scenario,

the multi-scale coordinates can be used to guide and maintain various objects spatial location at the global level, meanwhile, rich relevant relation information supports the model to distinguish boundaries of different objects at the local level. Thus, our model attains superior performance even with the simpler network structure. According to the experimental results, our model attains 8.6% and 7.2% performance gain on mIou when compared with PointNet++ and DGCNN under Area5 evaluation mode. Moreover, such superiority can also be observed on 6-fold cross-validation evaluation.

4.4. Ablation Analysis

Channel Number in Branches. Our model maintains a small constant channel number in four branches from end-to-end for efficiency purpose. We investigate how the channel number may affect the performance in this part in Table 8. Following the dual-space fusion learning architecture, our model achieves gratifying performance even with a very limited channel number (16). As observed, the model performance is gradually improved when the channel number is smaller than 64, which saturates once we adopt bigger value (128). Although bigger channel number may construct stronger model expression for semantic segmentation task, channel number fine-tuning is not pursued in this work. Thus, we simply use 64 in this work.

Table 8: Performance Comparison of different trunk channel number.

	ModelNet40			ShapeNetPart		
Channel Number	Input	mAcc	OA	Input	Cls. mIou	Ins. mIou
16	1k	89.8	92.6	2k	80.5	85.2
32	1k	90.2	92.9	2k	81.5	85.6
64	1k	**90.9**	**93.8**	2k	**82.8**	**86.6**
128	1k	90.7	93.0	2k	**82.8**	86.3

Global Information Guidance and Geometric Relation Supplement We conduct an ablation study to demonstrate the effectiveness of the two mechanisms in our geometric space learning. Table 9 demonstrates the obvious performance improvement of these two mechanisms on classification and part segmentation tasks. In addition, we observe that the contributions of these two mechanisms are different on classification and segmentation. Since information replenish is one of effective way to enhance model performance, our geometric relation supplement brings bigger accuracy improvement (0.8%) on relatively simple classification tasks. By introducing multi-scale object structure representations, geometric information guidance demonstrates equal significance for both tasks.

Table 9: Effectiveness demonstration of Global Information Guidance (GIG) and Geometric Relation Supplement (GRS) on ModelNet40 and ShapeNetPart.

		ModelNet40			ShapeNetPart		
GRS	GIG	mAcc	OA		Cls. mIou	Ins. mIou	
×	×	89.3	92.2		80.8	85.4	
✓	×	90.1	93.0	0.8 ↑	82.0	85.8	0.4 ↑
✓	✓	**90.9**	**93.8**	0.8 ↑	**82.8**	**86.6**	0.8 ↑

Geometric Relation Supplement Frequency Acknowledging that the structure relation is critical, we inject these spherical coordinate relation before each local feature updating process. In this part, We now examine the importance of geometric relation supplement frequency. Concretely, we evaluate the model performance when supplying the relevant relation only in the first local feature updating or in all local feature updating. By replenishing more geometric relation, PointGS achieves an improvement of 1.1% and 0.5% on ModelNet40 and ShapeNetPart respectively.

RGB Features Application By default, the RGB information is concatenated

Table 10: Ablation study of different geometric relation supplement frequency on ModelNet40 and ShapeNetPart.

	ModelNet40				ShapeNetPart			
Frequency	Input	mAcc	OA		Input	Cls. mIou	Ins. mIou	
1	1k	89.4	92.7		2k	82.6	86.1	
4	1k	**90.9**	**93.8**	1.1 ↑	2k	**82.8**	**86.6**	0.5 ↑

with coordinates to form input data. However, we argue that the physical meaning is irrelevant between spatial coordinates and color values. Instead of feeding them directly, we encode the RGB information into the Euclidean distance form as relevant feature relations. In this part, we investigate the model performance with three RGB information forms in S3DIS. The Area 5 evaluation results are listed in Table 11. As the mixed inputs may limit extracting distinct feature relation, the model performance is the worst once applying color values and the Euclidean distance forms together. By contrast, single data representation form is more effective, and the Euclidean distance form introduces more gains (2.6%) than adopting color values directly.

Table 11: Model performance with different RGB information forms on S3DIS Area 5. Dist means transforming color values to the Euclidean distance form, Value means inputting color values directly.

	S3DIS			
Model	OA	mAcc	mIou	
Dist and Value	86.1	65.3	56.5	
Value	86.0	68.1	59.1	2.6 ↑
Dist	**87.7**	**70.0**	**61.7**	2.6 ↑

4.5. Model Robustness and Efficiency

Model Robustness In the real scenes, we are also interested in ensuring that our method tolerates defective inputs. Here we plot models classification accuracy as the proportion of points dropped increases in Figure 5. It can be observed that PointNet++, DGCNN and our approach can retain high accuracy with 75% of points are randomly dropped. However, when over 75% points are dropped, the performance of PointNet++ and DGCNN degrades rapidly. On the contrary, our method could still attain over 60% accuracy even 95% points are dropped. Adversarial training strategy [56] and optimizer adjustment [57] are effective methods to further reinforce the robustness of models which will be attempted in our future works.

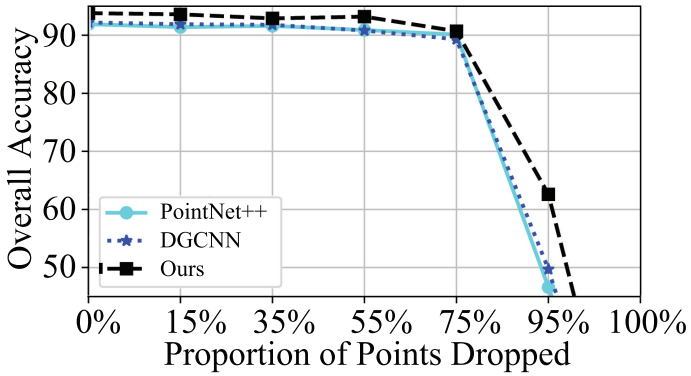

Figure 5: Robustness comparison on ModelNet40. Even if 95% of the points are randomly dropped, our method still obtains accuracy over 60%.

Model Efficiency We report the inference speed, i.e. throughput (samples/second), for our proposed PointGS in comparison with several models in Table 12. The numbers of input points in ShapeNetPart and S3DIS datasets are 2048 and 4098 respectively, and more input points lead to slower inference speed. For clarity, we report model inference speed on ModelNet40 dataset (1024 input points). All the comparison models were run on one Nvidia 3090Ti GPU and Inter(R) Xeon(R) Silver 3.20GHz CPU. The batch size is set to 10 and Pytorch 1.8 is applied. We report the average speed over 200 runs. As observed,

DGCNN and AdaptiveGraph avoid to use the Farthest Point Sampling (FPS) algorithm, which is inefficient on CPU. FPS is however necessary for PointNet++ and our method. Therefore, DGCNN and AdaptiveGraph may usually be faster than PointNet++ and our method. Compared with PointNet++, PointGS is marginally slower however leads to significantly higher accuracy than PointNet++. On the other hand, our proposed PointGS demonstrate higher or the same accuracy than PointConv and GDNet but exhibits much faster speed.

It is noted that there is still room to improve further the efficiency of the proposed PointGS. For example, some works [58, 18, 17] manage to compile the FPS algorithm by C++ for GPU implementation. Similarly, We implemented the FPS cuda implementation of [58] which proves able to improve substantially our model efficiency as seen in Table 12 (PointGS (Ours)*). We will leave the further exploration of our model's efficiency as future work.

Table 12: Inference speed comparison between PointNet++, DGCNN, AdaptiveGraph, and PointGS. We reporte the speed of some open sourced methods by samples/second tested on one NVIDIA 3090Ti GPU and Inter(R) Xeon(R) Silver 3.20GHz CPU. * means we adopt GPU implementation of FPS algorithms.

	ModelNet40		
Model	#Points	ThroughPut	OA
PointNet++ [9]	1k	68	90.7
DGCNN [10]	1k	2920	92.2
PointConv [34]	1k	5	92.5
AdaptiveGraph [24]	1k	1567	93.4
GDNet [59]	1k	7	**93.8**
PointGS (Ours)	1k	45	**93.8**
PointGS (Ours)*	1k	113	**93.8**

5. Visualization Analysis

In this part, we provide more typical visual illustration comparison to intuitively demonstrate the superiority of PointGS. With the mutual supervision mechanism on geometric and semantic spaces, PointGS enables to obtain stronger discriminative capability on different objects. Both the first and the second columns of Figure 6 show ground truth point cloud examples of ShapeNetPart dataset but in two different angles. For clarity, we adopt black circles to highlight the failure predictions of PointNet++ and DGCNN in the third and fourth columns, respectively. As observed, PointNet++ and DGCNN exploit simple network structures based on features from one single space; this lead to a lot of failures and limits their performance. As shown in Airplane3, Chair1, Motorbike, and Rocket, PointNet++ and DGCNN generate obvious prediction errors. In contrast, PointGS combines the strength of geometric structure and semantic representation and obtains more accurate results on these challenging objects. In addition, PointGS shows robust performance to handle even small parts segmentation (as observed in Airplane1,2 and Car1,2).

6. Conclusion

In this work, we explore a simple yet powerful architecture named PointGS for point cloud analysis. The key insight behind our method is that the single space learning may not be sufficient for better model performance. We propose to alternately learn in the geometric and semantic space to boost the performance. We first utilize FPS downsampling in the geometric space to form pyramid inputs which are helpful for rich global information extraction and avoids prohibitive computational pooling in the semantic space further. Then, we transform coordinates into semantic space and design local feature updating process for similar feature aggregation and features distillation. To reinforce the information fusion and interaction between two spaces, we iteratively perform dual-space fusion process which enables PointGS to establish a mutual supervision mechanism. Without sophisticated operations and elaborated feature

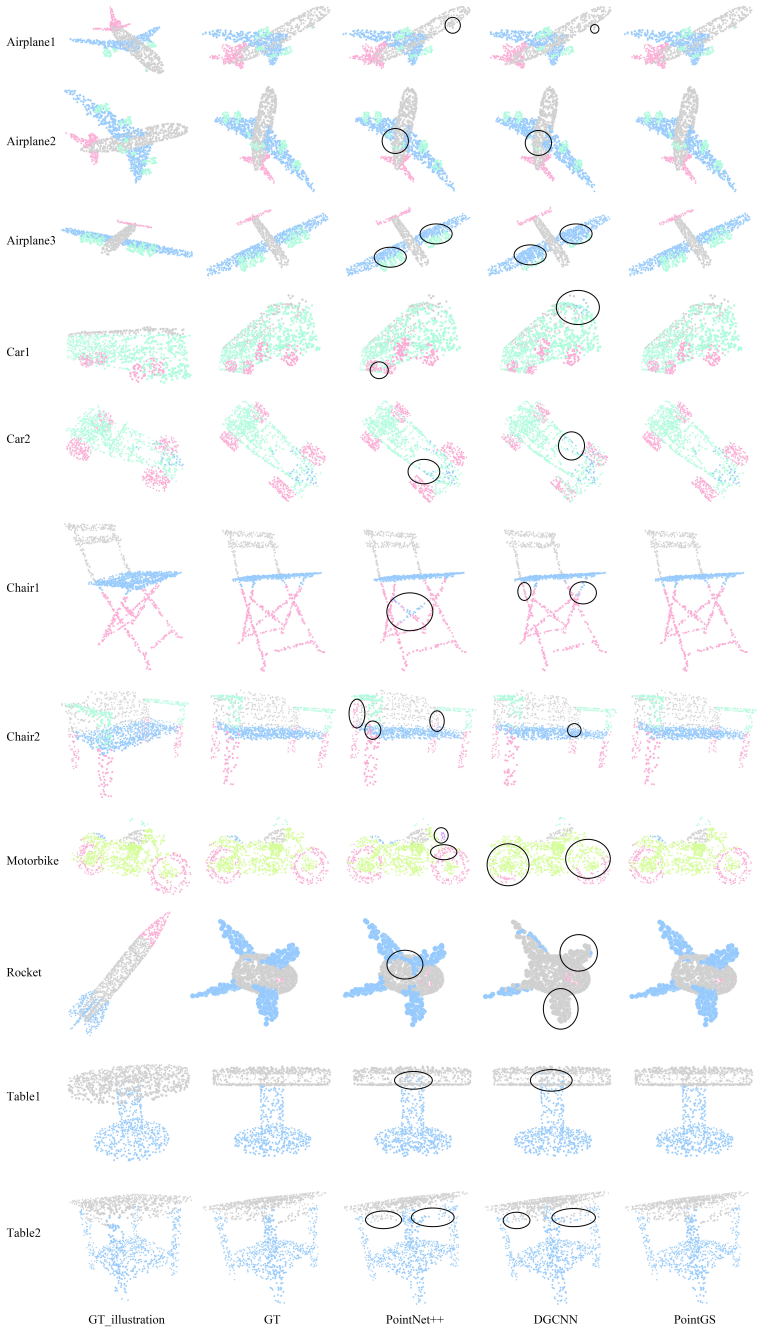

Figure 6: Visual comparison between PoitnNet++, DGCNN and PointGS. The first column provides intuitive viewpoint to demonstrate data. The third and fourth columns show prediction results of PointNet++ and DGCNN, and black circles denote the failure predictions of these two methods. Better to zoom in.

extractors, experimental results have shown that PointGS outperforms most popular works on different tasks. We hope that the work will provide a new sight for the community to rethink network design for point cloud analysis.

Acknowledgements

The work was partially supported by the following: National Natural Science Foundation of China under no.61876155; Natural Science Foundation of Jiangsu Province BE2020006-4, BK20181189; Key Program Special Fund in XJTLU under no. KSF-T-06, KSF-E-26.

References

[1] Y. An, J. Shi, D. Gu, Q. Liu, Visual-lidar slam based on unsupervised multi-channel deep neural networks, Cognitive Computation (2022) 1–13.

[2] T. Chen, D. Gu, Csa6d: Channel-spatial attention networks for 6d object pose estimation, Cognitive Computation 14 (2) (2022) 702–713.

[3] X. Chen, H. Ma, J. Wan, B. Li, T. Xia, Multi-view 3d object detection network for autonomous driving, in: CVPR, 2017.

[4] H. Su, S. Maji, E. Kalogerakis, E. Learned-Miller, Multi-view convolutional neural networks for 3d shape recognition, in: ICCV, 2015.

[5] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, O. Beijbom, Pointpillars: Fast encoders for object detection from point clouds, in: CVPR, 2019.

[6] D. Maturana, S. Scherer, Voxnet: A 3d convolutional neural network for real-time object recognition, in: IROS, 2015.

[7] G. Riegler, A. Osman Ulusoy, A. Geiger, Octnet: Learning deep 3d representations at high resolutions, in: CVPR, 2017.

[8] C. R. Qi, H. Su, K. Mo, L. J. Guibas, Pointnet: Deep learning on point sets for 3d classification and segmentation, in: CVPR, 2017.

[9] C. R. Qi, L. Yi, H. Su, L. J. Guibas, Pointnet++: Deep hierarchical feature learning on point sets in a metric space, NIPS.

[10] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, J. M. Solomon, Dynamic graph cnn for learning on point clouds, Acm Transactions On Graphics (tog) 38 (5) (2019) 1–12.

[11] D. Fernandes, A. Silva, R. Névoa, C. Simões, D. Gonzalez, M. Guevara, P. Novais, J. Monteiro, P. Melo-Pinto, Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy, Information Fusion 68 (2021) 161–191.

[12] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, L. J. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: ICCV, 2019.

[13] H. Zhao, L. Jiang, C.-W. Fu, J. Jia, Pointweb: Enhancing local neighborhood features for point cloud processing, in: CVPR, 2019.

[14] J. Mao, X. Wang, H. Li, Interpolated convolutional networks for 3d point cloud understanding, in: ICCV, 2019.

[15] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, B. Chen, Pointcnn: Convolution on x-transformed points, NIPS.

[16] S. Wang, S. Suo, W.-C. Ma, A. Pokrovsky, R. Urtasun, Deep parametric continuous convolutional neural networks, in: CVPR, 2018.

[17] G. Qian, Y. Li, H. Peng, J. Mai, H. A. A. K. Hammoud, M. Elhoseiny, B. Ghanem, Pointnext: Revisiting pointnet++ with improved training and scaling strategies, arXiv preprint arXiv:2206.04670.

[18] X. Ma, C. Qin, H. You, H. Ran, Y. Fu, Rethinking network design and local geometry in point cloud: A simple residual mlp framework, arXiv preprint arXiv:2202.07123.

[19] G. Qian, H. Hammoud, G. Li, A. Thabet, B. Ghanem, Assanet: An anisotropic separable set abstraction for efficient point cloud representation learning, Advances in Neural Information Processing Systems 34 (2021) 28119–28130.

[20] M. Simonovsky, N. Komodakis, Dynamic edge-conditioned filters in convolutional neural networks on graphs, in: CVPR, 2017.

[21] G. Li, M. Muller, A. Thabet, B. Ghanem, Deepgcns: Can gcns go as deep as cnns?, in: ICCV, 2019.

[22] L. Jiang, H. Zhao, S. Liu, X. Shen, C.-W. Fu, J. Jia, Hierarchical point-edge interaction network for point cloud semantic segmentation, in: ICCV, 2019.

[23] C. Wang, B. Samari, K. Siddiqi, Local spectral graph convolution for point set feature learning, in: ECCV, 2018.

[24] H. Zhou, Y. Feng, M. Fang, M. Wei, J. Qin, T. Lu, Adaptive graph convolution for point cloud analysis, in: ICCV, 2021.

[25] Q. Xu, X. Sun, C.-Y. Wu, P. Wang, U. Neumann, Grid-gcn for fast and scalable point cloud learning, in: CVPR, 2020, pp. 5661–5670.

[26] Z.-H. Lin, S.-Y. Huang, Y.-C. F. Wang, Learning of 3d graph convolution networks for point cloud analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence 44 (8) (2021) 4212–4224.

[27] T. Le, Y. Duan, Pointgrid: A deep network for 3d shape understanding, in: CVPR, 2018.

[28] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, X. Tong, O-cnn: Octree-based convolutional neural networks for 3d shape analysis, ACM Transactions On Graphics (TOG) 36 (4) (2017) 1–11.

[29] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, J. Xiao, 3d shapenets: A deep representation for volumetric shapes, in: CVPR, 2015.

[30] R. Klokov, V. Lempitsky, Escape from cells: Deep kd-networks for the recognition of 3d point cloud models, in: ICCV, 2017.

[31] E. Kalogerakis, M. Averkiou, S. Maji, S. Chaudhuri, 3d shape segmentation with projective convolutional networks, in: CVPR, 2017.

[32] Y. Feng, Z. Zhang, X. Zhao, R. Ji, Y. Gao, Gvcnn: Group-view convolutional neural networks for 3d shape recognition, in: CVPR, 2018.

[33] H. Guo, J. Wang, Y. Gao, J. Li, H. Lu, Multi-view 3d object retrieval with deep embedding network, IEEE Transactions on Image Processing.

[34] W. Wu, Z. Qi, L. Fuxin, Pointconv: Deep convolutional networks on 3d point clouds, in: CVPR, 2019.

[35] Y. Xu, T. Fan, M. Xu, L. Zeng, Y. Qiao, Spidercnn: Deep learning on point sets with parameterized convolutional filters, in: ECCV, 2018.

[36] L. Wang, Y. Huang, Y. Hou, S. Zhang, J. Shan, Graph attention convolution for point cloud semantic segmentation, in: CVPR, 2019.

[37] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, in: CVPR, 2018.

[38] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, S.-M. Hu, Pct: Point cloud transformer, Computational Visual Media 7 (2) (2021) 187–199.

[39] H. Zhao, L. Jiang, J. Jia, P. H. Torr, V. Koltun, Point transformer, in: ICCV, 2021.

[40] S. A. Tailor, R. de Jong, T. Azevedo, M. Mattina, P. Maji, Towards efficient point cloud graph neural networks through architectural simplification, in: ICCV, 2021.

[41] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE transactions on neural networks.

[42] S. Srivastava, G. Sharma, Exploiting local geometry for feature and graph construction for better 3d point cloud processing with graph neural networks, in: ICRA, 2021.

[43] Y. Shen, C. Feng, Y. Yang, D. Tian, Mining point cloud local structures by kernel correlation and graph pooling, in: CVPR, 2018.

[44] M. Atzmon, H. Maron, Y. Lipman, Point convolutional neural networks by extension operators, arXiv preprint arXiv:1803.10091.

[45] S. Qiu, S. Anwar, N. Barnes, Dense-resolution network for point cloud classification and segmentation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 3813–3822.

[46] X. Yan, C. Zheng, Z. Li, S. Wang, S. Cui, Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling, in: CVPR, 2020.

[47] J. Li, B. M. Chen, G. H. Lee, So-net: Self-organizing network for point cloud analysis, in: CVPR, 2018.

[48] W. Wang, R. Yu, Q. Huang, U. Neumann, Sgpn: Similarity group proposal network for 3d point cloud instance segmentation, in: CVPR, 2018.

[49] Y. Liu, B. Fan, S. Xiang, C. Pan, Relation-shape convolutional neural network for point cloud analysis, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8895–8904.

[50] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas, A scalable active framework for region annotation in 3d shape collections, ACM Transactions on Graphics (ToG) 35 (6) (2016) 1–12.

[51] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3d semantic parsing of large-scale indoor spaces, in: CVPR, 2016.

[52] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, S. Savarese, Segcloud: Semantic segmentation of 3d point clouds, in: 3DV, 2017.

[53] M. Tatarchenko, J. Park, V. Koltun, Q.-Y. Zhou, Tangent convolutions for dense prediction in 3d, in: CVPR, 2018.

[54] Q. Huang, W. Wang, U. Neumann, Recurrent slice networks for 3d segmentation of point clouds, in: CVPR, 2018.

[55] A. Komarichev, Z. Zhong, J. Hua, A-cnn: Annularly convolutional neural networks on point clouds, in: CVPR, 2019.

[56] H. Jiang, K. Huang, R. Zhang, A. Hussain, Style-neutralized pattern classification based on adversarially trained upgraded u-net, Cognitive Computation 13 (4) (2021) 845–858.

[57] Y. Zhou, K. Huang, C. Cheng, X. Wang, X. Liu, Lightadam: Towards a fast and accurate adaptive momentum online algorithm, Cognitive Computation 14 (2) (2022) 764–779.

[58] H. Ran, J. Liu, C. Wang, Surface representation for point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 18942–18952.

[59] M. Xu, J. Zhang, Z. Zhou, M. Xu, X. Qi, Y. Qiao, Learning geometry-disentangled representation for complementary understanding of 3d object point cloud, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 3056–3064.