PnP-3D: A Plug-and-Play for 3D Point Clouds

Shi Qiu, Saeed Anwar, and Nick Barnes

Abstract—With the help of the deep learning paradigm, many point cloud networks have been invented for visual analysis. However, there is great potential for development of these networks since the given information of point cloud data has not been fully exploited. To improve the effectiveness of existing networks in analyzing point cloud data, we propose a plug-and-play module, PnP-3D, aiming to refine the fundamental point cloud feature representations by involving more local context and global bilinear response from explicit 3D space and implicit feature space. To thoroughly evaluate our approach, we conduct experiments on three standard point cloud analysis tasks, including classification, semantic segmentation, and object detection, where we select three state-of-the-art networks from each task for evaluation. Serving as a plug-and-play module, PnP-3D can significantly boost the performances of established networks. In addition to achieving state-of-the-art results on four widely used point cloud benchmarks, we present comprehensive ablation studies and visualizations to demonstrate our approach's advantages. The code will be available at https://github.com/ShiQiu0419/pnp-3d.

Index Terms—Point Cloud, Plug-and-Play, Feature Representation, Classification, Segmentation, Detection, 3D Deep Learning.

1 Introduction

Point cloud data shows increasing value in both academia and industry, since affordable yet effective 3D sensors [1], [2] are widely used in autonomous driving, robotics, augmented reality, *etc.* However, unlike the 2D images that have well-organized structure, point cloud data is *unordered* and *unevenly distributed* in 3D space, increasing the difficulty of machine perception towards common visual tasks such as classification, segmentation, detection, *etc.*

Following the success of Convolutional Neural Networks (CNNs) in 2D images, early methods [3], [4] captured 2D multi-view projections of point clouds for 2D CNN processing. Then, to preserve the 3D context of raw data and avoid the conversion between 2D and 3D representations, researchers proposed point cloud networks such as [5], [6], [7] to directly process 3D data for visual analysis. However, recent research evidence suggests that the capability of existing networks has not been fully exploited: *e.g.*, the PointAugment framework [8] improves the performance of classification networks [9], [10] by automatically augmenting point cloud samples; and the CGA module [11] boosts the effectiveness of segmentation networks [12], [13] by organizing category-aware neighborhoods. Beyond such *task-specific* modules to facilitate the network's understanding of a certain task, we aim to explore a generic and effective *plug-and-play* module that can be easily deployed in different point cloud networks without additional setup. To achieve broader usage in various tasks, basically, we need to emphasize a fundamental component of point cloud analysis: point feature representation.

Importance of point feature representation. With the help of CNN's strong expression and generalization capacity, we represent point features in different dimensions of embedding space and from multiple point cloud resolutions. Then we can conduct various operations on the point feature representations to analyze some basic properties of point cloud data. To be specific, the classification task [8], [14] usually infers the point cloud's category based on its global embedding vector, which is aggregated from the entire feature map (*i.e.*, the feature representations of all points) via a global pooling function. Moreover, semantic segmentation networks [12], [15] directly apply shared fully connected layers to each point feature representation for its semantic label prediction. Similarly, the 3D detection pipeline [16] incorporates a point feature learning backbone generating votes for object proposals. In general, point cloud analysis significantly relies on the corresponding point feature representations; thus, for high-quality point cloud analysis, we can further *refine* the learned features. Apart from the attention mechanism [17] that implicitly re-weights the importance of each element in the feature map, first of all, we aim to explicitly enrich the local information of each point by incorporating different types of context.

Variety of point cloud context. Recently, an increasing number of methods [12], [15], [19] choose to directly combine the point cloud's inherent 3D coordinates (*i.e.*, the geometric context) with the point feature representations acquired by CNNs. The advantages can be summarized in two aspects: firstly, as the network goes deeper, the learned features in higher-dimensional embedding space will be more abstract and implicit, thus, we need to introduce the fundamental 3D geometry attribute for the network's reference; secondly, since such 3D coordinates can indicate the point distribution in the real-world, we can additionally gather more local information from the constructed point neighborhoods under a particular spatial metric such as 3D Euclidean distance in the ball query [6], [10] or the k-nearest-neighbors (knn) [9], [12] algorithm. However, repeatedly and directly feeding such 3D coordinates to the network is not always an ideal solution. Instead, we try to fuse the local *geometric* and *feature* context, which are captured in parallel following the same constraints and formulation derived from the point cloud's inherent 3D coordinates. By gathering the local information, not only do we introduce 3D geometric

- *S. Qiu and S. Anwar are with Data61, CSIRO (The Commonwealth Scientific and Industrial Research Organisation) and School of Engineering, Australian National University, Canberra, ACT 2601, Australia.*
- *N. Barnes is with School of Computing, Australian National University, Canberra, ACT 2601, Australia.*
- *E-mail: {shi.qiu, saeed.anwar, nick.barnes}@anu.edu.au*

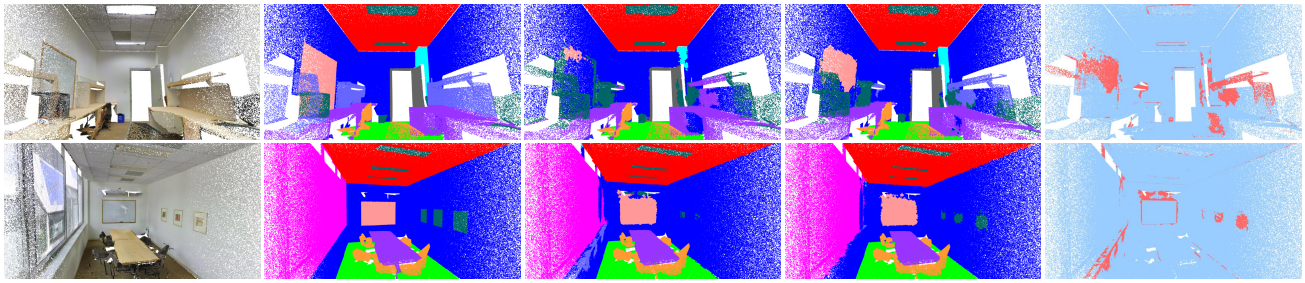


Fig. 1. Examples of point cloud semantic segmentation on *S3DIS* [18] dataset. The 3rd column shows the results of RandLA-Net [12] (baseline), while the 4th column presents the results of plugging our PnP-3D module in the baseline. The improved areas (*i.e.*, the points that are correctly classified by using our module but misclassified by the baseline) are highlighted with red color in the last column.

relationships for the network, but we also provide more local distinctness in point feature representation.

Global perception of point cloud data. Besides incorporating different types of local context, we still need to refine point feature representations from a global perspective (*w.r.t.* the entire feature map) for two main purposes: (i) to strengthen the inter-dependencies between point-wise and channel-wise elements; (ii) to regularize all the learned point features for the machine's explicit understanding. Although some previous methods [14], [19], [20] employed self-attention structures [21] to estimate element-wise dependencies, due to the complexity of point cloud data and networks, extra heavy computations for self-attention are not practically preferable. Alternatively, we can use lightweight operations to generate a global bilinear response consisting of both point-wise and channel-wise inter-dependencies. Coupled with rich local information, a comprehensive feature map can be synthesized for accurate point cloud analysis.

2 Related Work

Point Cloud Analysis Tasks. As a fundamental problem in point cloud analysis, the classification task aims to identify the category of a point cloud *object*. To deal with the unordered points, regular methods [5], [8], [14] apply a symmetric function (*e.g.*, max-pooling) to aggregate a global embedding from its feature map. Based on the global embedding, we further regress the possibilities regarding candidate categories via fully connected layers. For a more fine-grained analysis of a point cloud *scene*, we can segment the whole point cloud into a few clusters based on different types of point-related properties such as the object-part class [22], [23], semantic category [12], [15], or instance index [24], [25]. Particularly, the basic point cloud semantic segmentation relies highly on the point feature representation to predict each point's semantic label. Moreover, as autonomous driving techniques develop rapidly, point cloud object detection is becoming more popular in the 3D computer vision area. As the main track of solutions, the region proposal-based methods [16], [26], [27] utilize a backbone network learning the point feature representations and predict the 3D bounding boxes and categories of the physical objects based on the learned information.

To conclude, point feature representations are crucial in point cloud analysis. By using our *plug-and-play* module to synthesize a more comprehensive feature map, the effectiveness of point cloud networks can be further improved.

Point-based Networks. Unlike projection-based [3], [4] or discretization-based [28], [29] networks which *indirectly* analyze point cloud data, point-based networks [5], [6] are more widely adopted in point cloud analysis because of the intuitiveness and simplicity. To be specific, the basic point-based networks usually apply the Multi-Layer-Perceptron (MLP) [5] to *directly* learn the point-wise features from 3D point cloud data, while the advanced methods [6], [30] tend to form point neighborhoods for more local context. Later works extend the primary usage of MLPs in different ways: the graph-based methods [9], [23] extract local features from crafted point graphs; and the convolution-based approaches [10], [31], [32] develop several MLP variants by involving more geometric clues.

Even though most point-based networks learn the point feature representations from multiple point cloud resolutions, such a *plug-and-play* module can be easily and flexibly deployed to refine the feature map of each resolution, showing great adaptability in point-based networks.

Attention Mechanism for Point Clouds. As a powerful tool in deep learning, attention mechanisms [17] have been utilized to address many computer vision problems. Similar to the Non-local network [21] and SENet [33] for 2D images, we are able to re-weight the point cloud feature map by applying attention-based structures along either point-axis or channel-axis. To be concrete, [20] and [34] follow the self-attention pipeline to calculate the long-range dependencies between points, while GBNet [14] regresses the channel-wise affinities in a similar behavior. In addition, [35] and [36] leverage the idea of graph-attention [37] in graph-based point cloud networks. More recently, [38] and [39] achieve good performance on the point cloud classification and segmentation benchmarks by forming a transformer-based network [40], where the regular point feature encoders and decoders are replaced with the attention-based modules.

Given the complexity of point cloud data, we propose an effective *plug-and-play* module, exceeding the function of regular attention approaches in two ways: (i) more local information is incorporated to enrich the point context; and (ii) both point-wise and channel-wise global perceptions are exploited to regulate the feature representations.

3 Approach

3.1 Overview

Given a point cloud with N points, the 3D coordinates can be presented as $\mathcal{P} = [p_1; ...; p_i; ...; p_N] \in \mathbb{R}^{N\times 3}$, where

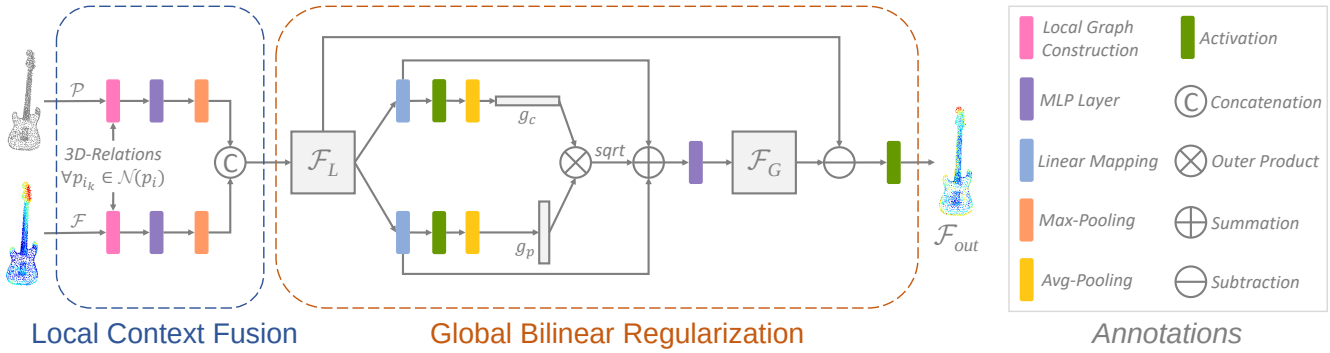


Fig. 2. Detailed structure of the PnP-3D module. The local context fusion block (Section 3.2) aims to fuse local geometric and feature context based on the geometric relations between points in 3D space. Further, the global bilinear regularization block (Section 3.3) can regularize the feature map by aggregating global perceptions based on both point-wise and channel-wise information in feature space. More implementation details are in Section 4.

an arbitrary point is denoted as $p_i \in \mathbb{R}^3$. Moreover, we can use CNN-based modules to learn the feature map $\mathcal{F} = [f_1; ...; f_i; ...; f_N] \in \mathbb{R}^{N \times C}$ in a C-dimensional space, where p_i's corresponding feature representation is $f_i \in \mathbb{R}^C$.

In language-related usage [17] of the attention mechanism, the positional encoding operation gives a cue for *sequence order* to the network by inputting the position of tokens. Similarly, in the case of unordered point cloud data, we need to inform the network about the *geometric relations* between scattered points. As shown in Figure 2, we deploy a local context fusion block to fulfill this intention.

3.2 Local Context Fusion

Following either a knn [9] or ball-query algorithm [6], we can simply find the neighbors $\forall p_{i_k} \in \mathcal{N}(p_i)$ of a certain point p_i, under a metric of *3D Euclidean distance* between the scattered points. By combining the edges [9] between p_i itself and its k neighbors $\forall p_{i_k} \in \mathcal{N}(p_i)$, we define a local geometric graph of p_i in 3D space follows: $\tilde{p_i} = [p_i; p_{i_k} - p_i]$, $\tilde{p_i} \in \mathbb{R}^{k \times 6}$. Thus, the local geometric graphs for all points $\mathcal{P}$ are generally denoted as $\tilde{\mathcal{P}} = [\tilde{p_1}; ...; \tilde{p_i}; ...; \tilde{p_N}] \in \mathbb{R}^{N \times k \times 6}$.

Further, we encode local geometric graphs via a shared MLP $\boldsymbol{\mathcal{M}}_\Theta$, and aggregate the local geometric context by applying the max-pooling function over the k neighbors:

$$\mathring{\mathcal{P}} = \max_k \left(\boldsymbol{\mathcal{M}}_\Theta(\tilde{\mathcal{P}})\right), \quad \mathring{\mathcal{P}} \in \mathbb{R}^{N \times \frac{C}{2}}; \tag{1}$$

practically we implement the shared MLP operation as a 1×1 convolution followed by a batch normalization (BN) and an activation layer.

In parallel, a local feature graph of f_i in C-dimensional space can be formed as: $\tilde{f}_i = [f_i; f_{i_k} - f_i] \in \mathbb{R}^{k \times 2C}$, where $\forall f_{i_k}$ are corresponding features of $\forall p_{i_k} \in \mathcal{N}(p_i)$. Accordingly, local feature graphs of the feature map $\mathcal{F}$ are represented as: $\tilde{\mathcal{F}} = [\tilde{f}_1; ...; \tilde{f}_i; ...; \tilde{f}_N] \in \mathbb{R}^{N \times k \times 2C}$. Following similar operations in Equation 1, we obtain the local feature context:

$$\mathring{\mathcal{F}} = \max_k \left(\boldsymbol{\mathcal{M}}_\Phi(\tilde{\mathcal{F}})\right), \quad \mathring{\mathcal{F}} \in \mathbb{R}^{N \times \frac{C}{2}}; \tag{2}$$

where $\boldsymbol{\mathcal{M}}_\Phi$ is another shared MLP encoding local feature graphs. Finally, we concatenate the local geometric context $\mathring{\mathcal{P}}$ and the local feature context $\mathring{\mathcal{F}}$ as the output of our local context fusion block:

$$\mathcal{F}_L = \text{concat}(\mathring{\mathcal{P}}, \mathring{\mathcal{F}}), \quad \mathcal{F}_L \in \mathbb{R}^{N \times C}. \tag{3}$$

Compared to the *EdgeConv* [9] operation, both the local geometric and feature context in our approach are formulated and constrained with inherent 3D geometric relations. Furthermore, the local context fusion block can be flexibly deployed at different point cloud resolutions and CNN layers, benefiting most of the existing point cloud networks.

3.3 Global Bilinear Regularization

In addition to gathering more local detail for each point's feature representation, the global bilinear regularization block aims to refine the feature map by taking global perceptions of the whole point cloud into account. Recall that in regular self-attention mechanisms, global perceptions are estimated as long-range dependencies (*i.e.*, cosine similarities) between point-wise features, consuming a relatively large amount of memory. Instead, global perceptions in our approach are computed as the feature map's element-level inter-dependencies based on the global channel-wise and point-wise descriptors.

To encode the global channel-wise descriptor: first, we apply a weight matrix, $\boldsymbol{W_c} \in \mathbb{R}^{C \times \frac{C}{r}}$ where r is a reduction factor, to reduce the dimension of the fusion output $\mathcal{F}_L \in \mathbb{R}^{N \times C}$; then, we leverage the ReLU [41] function to not only provide non-linearity after the linear mapping with $\boldsymbol{W_c}$, but also meet the demand of non-negativity in Equation 6; finally, by conducting the average-pooling operation over the N elements along space-axis, we can squeeze the spatial information [33] as a global channel-wise descriptor g_c. Mathematically, the above operations follow:

$$g_c = \underset{N}{\text{avg}}\big(\text{ReLU}(\mathcal{F}_L \boldsymbol{W_c})\big), \quad g_c \in \mathbb{R}^{\frac{C}{r}}; \tag{4}$$

where "$\mathcal{F}_L \boldsymbol{W_c}$" is the *matrix product* between $\mathcal{F}_L$ and $\boldsymbol{W_c}$, "avg" is the average-pooling, and the reduction factor (an integer) satisfies: $r \geq 2$. Further, $g_c = [\mu_1, ..., \mu_j, ..., \mu_{\frac{C}{r}}]$, and $\mu_j \in \mathbb{R}$ stands for the global (mean) response of the j-th *channel* in a feature map of whole point cloud.

With another weight matrix $\boldsymbol{W_p} \in \mathbb{R}^{C \times \frac{C}{r}}$, ReLU function, and the average-pooling operation over the $\frac{C}{r}$ elements along channel-axis, we can also generate a global point-wise descriptor g_p in a similar way to that used in Equation 4:

$$g_p = \underset{\frac{C}{r}}{\text{avg}}\big(\text{ReLU}(\mathcal{F}_L \boldsymbol{W_p})\big), \quad g_p \in \mathbb{R}^N, \tag{5}$$

where $g_p = [\lambda_1, ..., \lambda_i, ..., \lambda_N]$ and $\lambda_i \in \mathbb{R}$ estimate the global (mean) response of the i-th *point* in the whole point cloud's feature map; "$\mathcal{F}_L \boldsymbol{W_p}$" is another *matrix product*.

Unlike [42] which uses a Hadamard product (*i.e.*, element-wise product) between the vectors, we capture a low-rank global bilinear response by taking the square root of g_p and g_c's *outer* product:

$$\mathcal{G} = \mathrm{sqrt}(g_p \otimes g_c), \quad \mathcal{G} \in \mathbb{R}^{N \times \frac{C}{r}}; \tag{6}$$

where an element η_{ij} positioned at the i-th row and j-th column of $\mathcal{G}$ is mathematically calculated as:

$$\eta_{ij} = \sqrt{\lambda_i \mu_j}, \quad \eta_{ij} \in \mathbb{R}. \tag{7}$$

The reasons of using a global bilinear response can be explained from two aspects. Firstly, given that a global channel-wise descriptor captures channel-wise dependencies [33] and a global point-wise descriptor represents overall shape context [5], calculating a bilinear combination of these two global descriptors can comprehensively synthesize and fully preserve the corresponding two types of global information. Secondly, in terms of each element, as λ_i and μ_j are the *arithmetic* means of the i-th point and j-th channel respectively, η_{ij} is the *geometric mean* of λ_i and μ_j: it provides a higher-order mean response based on both spatial and channel-related information.

After applying a shared MLP $\boldsymbol{\mathcal{M}}_\Psi$ and two shortcut connections as suggested in [42], we finally restore the channel dimension, and generate a full-sized global perception map:

$$\mathcal{F}_G = \boldsymbol{\mathcal{M}}_\Psi(\mathcal{G} + \mathcal{F}_L \boldsymbol{W_c} + \mathcal{F}_L \boldsymbol{W_p}); \tag{8}$$

where $\mathcal{F}_G \in \mathbb{R}^{N \times C}$. Next, we expect to find a better usage of the global perception map.

The average-pooling operation in Equations 4 and 5 is used to squeeze the global information from the point space and channel space respectively, where the pooled mean values usually represent the *common* patterns [43], [44] in feature maps. However, in point cloud analysis, we would like the learned features to be more distinct and representative. In this case, to sharpen the learned features, the global perception $\mathcal{F}_G$ that is derived from two global mean vectors, g_p and g_c, can be further filtered out. In practice, we achieve this by subtracting $\mathcal{F}_G$ from the local context fusion output $\mathcal{F}_L$, and use an activation σ to add more non-linearity in the final output feature map:

$$\mathcal{F}_{out} = \sigma(\mathcal{F}_L - \mathcal{F}_G), \quad \mathcal{F}_{out} \in \mathbb{R}^{N \times C}. \tag{9}$$

As the selection of pooling and regularization strategy is always an open but practical question, we conduct the related ablation studies to investigate the best form of our global bilinear regularization block. More details and discussions can be found in Section 5.4.

4 Implementation Details

Since we mainly propose a *plug-and-play* module for point cloud analysis networks, for fair comparisons, most of the hyperparameters (*e.g.*, feature dimensions, training paradigms, pre/post-processing *etc.*) in our experiments are adopted from the baseline's implementation if not explicitly mentioned. In general, the PnP-3D module would be placed after each feature encoding layer (encoder) of a baseline network; more concrete settings are provided in Section 5.

As for the local context fusion block, the neighbors are searched and collected in the same way as in the baseline network. Particularly, in the point cloud semantic segmentation task, we only take half of the original k neighbors as suggested in [11] to save memory. Further, to minimize the module's complexity, all shared MLPs are implemented as the composition of a *single-layer* 1×1 convolution, a BN layer, and a ReLU activation.

In terms of the global bilinear regularization block, the matrix multiplication operation in Equation 4 and 5 is realized as a *single-layer* 1×1 convolution, which can linearly map the features into a lower-dimensional space. Empirically, we set the reduction factor $r = 8$ in all cases. Moreover, a self-regularized and non-monotonic activation function, Mish [45], is leveraged as σ in Equation 9.

TABLE 1
Overall classification results (%) on *ModelNet40* [46] and *ScanObjectNN* [47]. ("OA": Overall Accuracy; "mAcc": mean Class Accuracy; **Bold** numbers indicate the results higher than corresponding baselines. For each column, the *highest* value is highlighted in red.)

Method	*ModelNet40*		*ScanObjectNN*	
	OA	mAcc	OA	mAcc
PointNet [5]	89.2	86.0	68.2	63.4
PointCNN [48]	92.2	88.1	78.5	75.1
SpiderCNN [31]	92.4	-	73.7	69.8
DRNet [23]	93.1	-	80.3	78.0
RS-CNN [10]	92.2	88.3	75.2	71.6
+PnP-3D	**93.1**	**89.5**	**77.9**	**73.6**
PointNet++ [6]	90.7	87.6	77.9	75.4
+PnP-3D	**93.2**	**91.1**	**82.2**	**79.6**
DGCNN [9]	92.2	90.2	78.1	73.6
+PnP-3D	**93.4**	**90.8**	**81.0**	**78.0**
PAConv [49]	93.6	-	77.1	74.4
+PnP-3D	**93.8**	**91.4**	**80.3**	**76.8**
AdaptConv [50]	93.4	90.7	79.3	76.0
+PnP-3D	**93.8**	**91.1**	**81.5**	**78.9**

5 Experiments

To comprehensively validate the effectiveness of our approach in point cloud analysis, we conduct a wide range of experiments, including point cloud classification, semantic segmentation, and object detection using different baselines and datasets. In each of the following subsections, we provide the experimental details for a specific task.

5.1 Point Cloud Classification

Classification Baselines. As an extension of PointNet [5], PointNet++ [6] additionally encodes each point feature from a ball-like local area found by the ball-query algorithm. A U-Net shape investigates lower point cloud resolutions and then higher resolutions. In a similar architecture, RS-CNN [10] replaces the regular MLP with the relation shape convolution (RS-Conv) in PointNet++. On the contrary, DGCNN [9] gradually represents the full-sized point cloud feature maps using cascaded encoders, by which the point-level graphs can be dynamically crafted. Moreover, the latest PAConv [49] and AdaptConv [50] networks are also tested.

Classification Dataset. ModelNet40 [46] is a synthetic dataset made up of 12,311 CAD-generated point cloud

TABLE 2
Detailed semantic segmentation results (Intersection-over-Union, %) on the *Area 5* of *S3DIS* [18] dataset. ("mIoU": mean Intersection-over-Union; **Bold** numbers indicate the results higher than corresponding baselines. For each column, the *highest* value is highlighted in red.)

Method	mIoU	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [5]	41.1	88.8	97.3	69.8	0.1	3.9	46.3	10.8	52.6	58.9	40.3	5.9	26.4	33.2
PointCNN [48]	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 [51]	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
PointWeb [52]	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
PointASNL [19]	62.6	94.3	98.4	79.1	0.0	26.7	55.2	66.2	83.3	86.8	47.6	68.3	56.4	52.1
Minkowski-Net [28]	65.4	91.8	98.7	86.2	0.0	34.1	48.9	62.4	81.6	89.8	47.2	74.9	74.4	58.6
BAAF-Net [15]	65.4	92.9	97.9	82.3	0.0	23.1	65.5	64.9	78.5	87.5	61.4	70.7	68.7	57.2
KPConv [32]	67.1	92.6	97.3	81.4	0.0	16.5	54.5	69.5	90.1	80.2	74.6	66.4	63.7	58.1
JSENet [53]	67.7	93.8	97.0	83.0	0.0	23.2	61.3	71.6	89.9	79.8	75.6	72.3	72.7	60.4
PointNet++ [6]	53.5	89.4	97.7	75.4	0.0	1.8	58.3	19.5	69.2	79.0	46.2	59.1	58.7	41.6
+PnP-3D	**58.7**	**90.8**	**98.1**	**77.0**	0.0	**7.6**	**61.7**	**28.1**	**74.2**	**84.3**	**67.3**	**64.8**	**60.5**	**48.0**
RandLA-Net [12]	62.5	92.3	97.7	80.5	0.0	20.9	62.0	35.3	77.7	86.8	74.7	68.8	65.0	50.8
+PnP-3D	**65.7**	**92.5**	97.6	**82.0**	0.0	**34.4**	**64.0**	**52.1**	**78.5**	86.8	**75.4**	**70.0**	**69.5**	**51.8**
SCF-Net [54]	63.1	90.4	97.0	80.6	0.0	19.3	57.2	44.8	78.6	86.9	75.9	71.4	68.4	49.9
+PnP-3D	**65.8**	**91.2**	**97.1**	**82.0**	0.0	**26.9**	**62.4**	**51.9**	78.6	**88.6**	**80.2**	**71.6**	**73.0**	**51.4**
CloserLook3D [13]	65.7	93.9	98.3	82.0	0.0	18.2	56.6	68.0	91.2	80.3	75.3	58.4	70.6	60.8
+PnP-3D	**68.5**	**94.7**	**98.4**	**83.7**	0.0	**21.1**	**60.4**	64.4	**92.9**	**83.1**	**76.8**	**83.5**	69.1	**62.2**

samples in 40 different object categories. In particular, the corresponding point cloud data is uniformly sampled from the surface of each mesh. Following the official data split, we have 9,843 point clouds in the training set, while the remaining 2,468 point clouds are for testing. In addition, ScanObjectNN [47] has around 14,298 point clouds (11,416 training samples and 2,882 testing samples) in 15 classes, which are manually scanned from a real-world environment. As suggest in [47], we use the hardest perturbation variant of ScanObjectNN for our experiments.

Experimental Settings. In general, we take the 3D coordinates of 1024 points as each point cloud's input for all classification experiments. Specifically, for both RS-CNN and PointNet++, the single-scale model is deployed as our baseline, where the PnP-3D module is placed after each Set Abstraction (SA) layer. For the rest baselines, we apply our module to refine each output of the convolution block.

Classification results. Table 1 presents the detailed classification results on ModelNet40 and ScanObjectNN datasets. Using the PnP-3D module, all five tested baseline networks achieve state-of-the-art performances (over 93% in overall accuracy) on ModelNet40 under the most basic input condition (3D coordinates of 1024 points). Further, after deploying the PnP-3D module, all baselines obtain significant improvements (2.2% to 4.3%) on ScanObjectNN compared with their original results. Particularly, as the most widely used baseline, PointNet++ shows great potential with the help of our approach beating all methods listed on ScanObjectNN leaderboard[1].

5.2 Point Cloud Semantic Segmentation

Semantic Segmentation Baselines. In order to verify the effectiveness of our module on the point cloud semantic segmentation task, besides PointNet++, we adopt another three recently introduced networks. Specifically, both RandLA-Net [12] and SCF-Net [54] take advantage of Random Sampling and Nearest Neighbor Interpolation methods to boost the efficiency in processing large-scale point cloud data, while CloserLook3D [13] involves different local aggregation operators in a typical ResNet [55] architecture.

1. https://hkust-vgd.github.io/scanobjectnn/

Semantic Segmentation Dataset. S3DIS [18] is precisely scanned from 6 main indoor working areas containing a total of 272 rooms in different types such as office, storage or conference room, *etc.* In particular, each room's point cloud data is made up of 0.5 to 2.5 million points labeled in 13 semantic classes. In the experiment, we take Area 5 as the test set while the remaining five areas are reserved for training. For fair comparisons, the RGB colors of points are used as additional input information.

Experimental Settings. Similarly, as in Section 5.1, the PnP-3D module is placed behind the SA layer in single-scale PointNet++ and the encoder in RandLA-Net and SCF-Net. As for CloserLook3D, our module refines the output feature map in every stage of its backbone (width=144, repeating factor=1, bottleneck ratio=2). Additionally, to testify to our module's effects on refining *pseudo grid* based features that are heavily exploited in [28], [32], [48], we use the *pseudo grid* local aggregation operator in CloserLook3D baseline.

Semantic Segmentation results. As shown in Table 2, the PnP-3D module can significantly boost the baseline's overall performances (by 2.7% to 5.2% mIoU) on the S3DIS dataset. Most of the 13 semantic categories can achieve varying degrees of improvement for each baseline by using our module. In terms of the categories like *column, door, sofa, and bookcase*, the corresponding IoUs show substantial growth in all networks, where the highest growth is more than 25%. In particular, we successfully achieve the highest mIoU, 68.5%, on Area 5 test set by utilizing the PnP-3D module in the CloserLook3D baseline.

5.3 Point Cloud Object Detection

Object Detection Baselines. VoteNet [16] is an end-to-end generic 3D point cloud object detection network. With the point features learned from the backbone, VoteNet generates some votes indicating the directions to the centers of candidate objects, then predicts semantic labels and 3D bounding boxes based on the votes. As a simplified version, BoxNet [16] directly makes the predictions based on the backbone's output without such a voting process. In contrast, ImVoteNet [26] extends the pipeline of VoteNet by lifting additional 2D votes in images for 3D votes in the point cloud.

TABLE 3
Detailed object detection results (Average Precision, %) on the *validation set* of *SUN RGB-D V1* [56] dataset. ("mAP": mean AP, IoU threshold of 0.25 [56]; **Bold** numbers indicate the results higher than corresponding baselines. For each column, the *highest* value is highlighted in red.)

Method	mAP	bath tub	bed	book shelf	chair	desk	dresser	night stand	sofa	table	toilet
DSS [57]	42.1	44.2	78.8	11.9	61.2	20.5	6.4	15.4	53.5	50.3	78.9
COG [58]	47.6	58.3	63.7	31.8	62.2	45.2	15.5	27.4	51.0	51.3	70.1
2D-driven [59]	45.1	43.5	64.5	31.4	48.3	27.9	25.9	41.9	50.4	37.0	80.4
F-PointNet [60]	54.0	43.3	81.1	33.3	64.2	24.7	32.0	58.1	61.1	51.1	90.9
MLCVNet [27]	59.8	79.2	85.8	31.9	75.8	26.5	31.3	61.5	66.3	50.4	89.1
MLCVNet++ [61]	60.9	79.3	85.3	36.5	77.1	28.7	31.6	61.4	68.3	50.7	90.0
BoxNet [16]	52.4	63.4	85.4	33.4	70.9	20.2	22.6	38.5	65.8	41.1	83.3
+PnP-3D	**54.1**	**73.5**	83.7	**34.3**	69.9	**20.6**	21.2	**44.2**	63.0	**41.6**	**88.6**
VoteNet [16]	57.7	74.4	83.0	28.8	75.3	22.0	29.8	62.2	64.0	47.3	90.1
+PnP-3D	**59.1**	**74.8**	**84.7**	**32.1**	75.0	**26.4**	28.8	**66.0**	**64.2**	**49.3**	89.4
ImVoteNet [26]	62.3	70.5	87.8	41.9	75.6	27.6	39.9	67.0	70.7	50.2	91.6
+PnP-3D	**63.8**	**74.9**	**88.6**	**42.9**	**76.3**	**28.1**	**41.7**	**69.5**	70.5	**52.2**	**92.7**

TABLE 4
Selecting the pooling and regularization strategy, tested on RandLA-Net [12] and *Area 5* of *S3DIS* [18] dataset. ("$\odot$", "$\oplus$", "$\ominus$": element-wise product/summation/subtraction.)

Model	Operation (in Equation 4)	Operation (in Equation 5)	Regularization (in Equation 9)	mIoU
1	max-pooling	max-pooling	$\odot$	63.0
2	max-pooling	max-pooling	$\oplus$	64.3
3	max-pooling	max-pooling	$\ominus$	63.3
4	avg-pooling	avg-pooling	$\odot$	63.7
5	avg-pooling	avg-pooling	$\oplus$	64.5
6	**avg-pooling**	**avg-pooling**	$\ominus$	**65.5**

TABLE 5
Generating the global bilinear response (η_{ij}) with point-wise response (λ_i) and channel-wise response (μ_j), tested on RandLA-Net [12] and *Area 5* of *S3DIS* [18] dataset.

Model	Meaning of η_{ij}	Formula (in Equation 6)	mIoU
1	Sum	$\lambda_i + \mu_j$	64.7
2	Product	$\lambda_i \mu_j$	64.2
3	Grand Mean	$(\lambda_i + \mu_j)/2$	63.7
4	Quadratic Mean	$\mathrm{sqrt}({\lambda_i}^2 + {\mu_j}^2)$	64.4
5	Harmonic Mean	$(2\lambda_i \mu_j)/(\lambda_i + \mu_j)$	65.2
6	**Geometric Mean**	$\mathrm{sqrt}(\lambda_i \mu_j)$	**65.5**

Object Detection Dataset. SUN RGB-D [56] is a single-view RGB-D dataset, which contains 5285 samples for training and 5050 testing examples. Based on the provided camera parameters, the corresponding 3D point cloud data can be generated. To fairly compare with the results in [16], [26], we only use the 3D coordinates as our input. Following the evaluation protocol in [16], [26], we report the ten most common object categories in the dataset.

Experimental Settings. As the point cloud object detection pipeline often takes a backbone to generate the seed point features, it is better to use our proposed module in the backbone rather than the voting process. In practice, since all the backbones in the three baseline networks are built with SA and FP layers of PointNet++, we also place the PnP-3D module after each SA layer as in Section 5.1 and 5.2.

Object Detection results. Table 3 clearly indicates the increasing margin caused by using the PnP-3D module in the baseline's backbone. Concretely, our approach benefits the categories of *bathtub, bookshelf, desk, nightstand, table* in all baselines, where the highest growth is over 10%. Although the overall improvements (by 1.4% to 1.7% mAP) are not as significant as the ones in Table 1 and 2 since only the backbone is modified, we still achieve a state-of-the-art result (63.8% mAP) on the SUN RGB-D dataset with ImVoteNet [26] baseline.

5.4 Ablation Studies

Pooling and Regularization Strategy. In addition to the average-pooling operation used in Equation 4 and 5, alternatively, the max-pooling operation can extract the *prominent* features to regularize (Equation 9) the feature map $\mathcal{F}_L$ learned from our local context fusion block. To investigate a best combination, we conduct experiments using different pooling (*i.e.*, max/average-pooling) and regularization (*i.e.*, element-wise product/summation/subtraction) strategies. The result of model 6 in Table 4 indicates that filtering out (*i.e.*, subtracting) the *common* features (*i.e.*, extracted by the average-pooling operation) from $\mathcal{F}_L$ is more effective.

Global Response Generation. Another set-up that needs further investigation is the way of generating a global response η_{ij} using both point-wise response λ_i and channel-wise response μ_j. Besides Equation 6 calculating the *geometric mean* of λ_i and μ_j, we explore other possible usages in Table 5. The experimental results show that a higher-order mean value (*e.g.*, quadratic/harmonic/geometric mean) can better estimate a global response for spatial and channel-related information, where the geometric mean (model 6) achieves the highest performance.

Comparisons with Attention Modules. As introduced in Section 1 and 2, the attention mechanism can also be applied in point cloud networks with the intention of refining the feature map. To this end, we compare the effectiveness of the PnP-3D module with *five* recent 3D attention approaches to point cloud object detection. According to the results shown in Table 6, our approach outperforms the competitors under the mAP metric in both IoU thresholds of 0.25 and 0.5.

In particular, we notice that some attention approaches [20], [34], [39] cannot benefit the baseline network [16] since they excessively exploit one type of information (*e.g.*, point-wise) causing redundancies in the feature representations. Instead, the PnP-3D module captures both local and global context following a compact design, balancing the effectiveness and computational cost. As shown in Table 6, our method costs the fewest parameters while achieving the highest performance. Compared to the latest

TABLE 6
Comparisons with the attention modules using VoteNet [16] on *SUN RGB-D V1* [56] dataset.("Dual": from both point-axis and channel-axis; "Local": from local areas; the value behind "@": IoU threshold.)

Method	Additional Info Origin	Model Size (MB)	Parameters ($\times 10^6$)	FLOPs ($\times 10^9$)	mAP (%) @0.25	mAP (%) @0.5
baseline [16]	–	10.9	0.95	48.5	57.7	33.1
+A-SCN [20]	Point-axis	15.9	1.39	**48.8**	55.6	30.1
+Point-attn [34]	Point-axis	15.9	1.39	**48.8**	56.4	32.2
+CAA [14]	Channel-axis	34.7	3.03	50.3	58.8	33.3
+Offset-attn [39]	Point-axis	19.5	1.73	50.4	55.7	30.6
+Point-Trans [38]	Dual + Local	25.7	2.24	126.0	58.1	34.8
+PnP-3D	Dual + Local	**14.4**	**1.25**	50.2	**59.1**	**34.9**

TABLE 7
Comparisons with the task-specific modules using different baselines, tested on *ModelNet40* [46] (classification), *S3DIS* [18] (semantic segmentation) and *SUN RGB-D V1* [56] (object detection) datasets.

	Method	OA	+PA [8]	+PnP-3D
Classification	PointNet++ [6]	90.7	92.9	**93.2**
	DGCNN [9]	92.2	**93.4**	**93.4**
	RS-CNN [10]	92.2	92.7	**93.1**
	method	mIoU	+CGA [11]	+PnP-3D
Segmentation	RandLA-Net [6]	62.5	65.4	**65.7**
	CloserLook3D [13]	65.7	**68.6**	68.5
	method	mAP	+RGB color	+PnP-3D
Detection	VoteNet [16]	57.7	56.3	**59.1**
	ImVoteNet [26]	62.3	63.4	**63.8**

Point-Transformer [38], our PnP-3D shows much higher efficiency *w.r.t.* the number of FLOPs.

Comparisons with Task-Specific Modules. We also compare our approach with different task-specific modules in all three tasks. As Table 7 indicates, the PnP-3D module provides higher improvements than the Point-Augment [8] framework in classification; while in semantic segmentation, we achieve the comparable results as using the CGA [11] module. Although there is no such a plug-and-play module for object detection, our approach can better benefit the baselines than feeding additional color information. Overall, our proposed plug-and-play module is more effective and generic for point cloud analysis tasks.

5.5 Visualizations

To analyze the PnP-3D module's behavior, we visualize and compare the feature maps in Figure 3. Generally, we observe that our approach can raise the high responses in more representative areas (*e.g.*, the wings/tail of plane, the armrest/leg of sofa/chair) covering a complete outline of point cloud object, while the baseline method (DGCNN [9]) only focuses on the central parts. This advantage can be credited to the global bilinear regularization block, which sharpens the feature map by integrating more global information based on both spatial and channel-related clues. Further, this property also benefits the semantic segmentation task as shown in Figure 1, where the number of "confusing" points (*i.e.*, near the boundaries of different categories) has been remarkably reduced.

In addition, we compare the generated votes for object detection in Figure 4. Recall that in the VoteNet [16] baseline, the votes are generated from the backbone's output feature map to estimate the centroids of detected objects. By leveraging the PnP-3D module in the backbone, more votes can be closely attached to the centroids (*e.g.*, shown as the right object in the top-right subfigure and the middle object in the bottom-right subfigure of Figure 4), providing more confident estimations for object proposals.

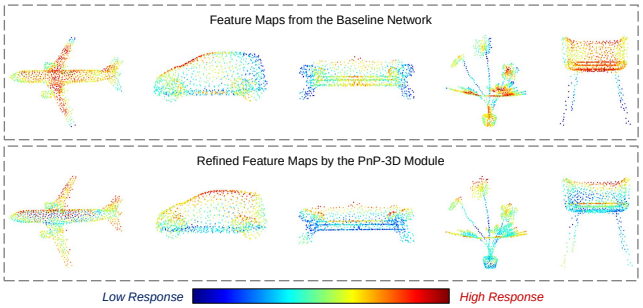
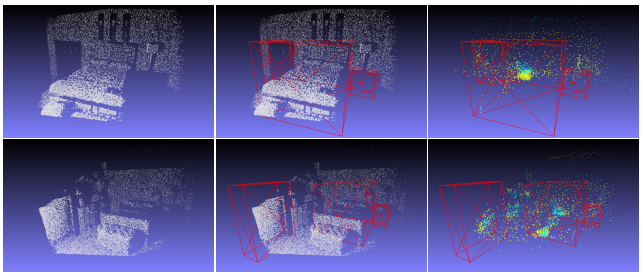


Fig. 3. Visualization of the feature maps in *ModelNet40* [46] classification. The first row shows the features learned from DGCNN [9], while the second row is the refined output from the PnP-3D module. We normalize the channels for a heat-map view. It can be clearly observed that our module better illustrates the outlines of point cloud objects.



(a) Input (b) Ground-truth (c) Generated votes

Fig. 4. Examples of the generated votes in *SUN RGB-D* [56] object detection. The bounding boxes (ground-truth) are in red frames, where the red points are the centroids. In the last column, we can find that the votes generated by the PnP-3D (blue points) better approach the object centroids than the baseline's outputs (VoteNet [16], yellow points).

6 Conclusion

In this paper, we focus on how to better refine the feature representations of point cloud data with a simple *plug-and-play*. To address this fundamental problem in point cloud analysis, we propose an effective module named PnP-3D, consisting of two specifically designed blocks. Concretely, the local context fusion block can fuse both local geometric and feature context according to the inherent point distribution in 3D space. Moreover, the global bilinear regularization block is leveraged to regularize the features by aggregating the global responses based on both point-wise and channel-wise information in feature space. By utilizing our module in different baselines and various point cloud analysis tasks, we comprehensively demonstrate the effectiveness of the PnP-3D module. The ablation studies and visualizations further verify the properties and behavior of our approach. In the future, we expect to extend its usage in low-level vision tasks such as point cloud upsampling or completion, and optimize its efficiency for real-time applications.

References

[1] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-d mapping with an rgb-d camera," *Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, 2013.
[2] M. Jaboyedoff, T. Oppikofer, A. Abellán, M.-H. Derron, A. Loye, R. Metzger, and A. Pedrazzini, "Use of lidar in landslide investigations: a review," *Natural hazards*, vol. 61, no. 1, pp. 5–28, 2012.
[3] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *ICCV*, 2015, pp. 945–953.

[4] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, "Deep projective 3d semantic segmentation," in *CAIP*, 2017, pp. 95–107.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *CVPR*, 2017, pp. 652–660.

[6] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NIPS*, 2017, pp. 5099–5108.

[7] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3d point clouds: A survey," *TPAMI*, 2020.

[8] R. Li, X. Li, P.-A. Heng, and C.-W. Fu, "Pointaugment: an auto-augmentation framework for point cloud classification," in *CVPR*, 2020, pp. 6378–6387.

[9] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *TOG*, vol. 38, no. 5, p. 146, 2019.

[10] Y. Liu, B. Fan, S. Xiang, and C. Pan, "Relation-shape convolutional neural network for point cloud analysis," in *CVPR*, 2019, pp. 8895–8904.

[11] T. Lu, L. Wang, and G. Wu, "Cga-net: Category guided aggregation for point cloud semantic segmentation," in *CVPR*, 2021, pp. 11 693–11 702.

[12] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham, "Randla-net: Efficient semantic segmentation of large-scale point clouds," in *CVPR*, 2020, pp. 11 108–11 117.

[13] Z. Liu, H. Hu, Y. Cao, Z. Zhang, and X. Tong, "A closer look at local aggregation operators in point cloud analysis," in *ECCV*. Springer, 2020, pp. 326–342.

[14] S. Qiu, S. Anwar, and N. Barnes, "Geometric back-projection network for point cloud classification," *IEEE TMM*, 2021.

[15] ——, "Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion," in *CVPR*, 2021, pp. 1757–1767.

[16] C. R. Qi, O. Litany, K. He, and L. J. Guibas, "Deep hough voting for 3d object detection in point clouds," in *ICCV*, 2019, pp. 9277–9286.

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[18] I. Armeni, S. Sax, A. R. Zamir, and S. Savarese, "Joint 2d-3d-semantic data for indoor scene understanding," *arXiv:1702.01105*, 2017.

[19] X. Yan, C. Zheng, Z. Li, S. Wang, and S. Cui, "Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling," in *CVPR*, 2020, pp. 5589–5598.

[20] S. Xie, S. Liu, Z. Chen, and Z. Tu, "Attentional shapecontextnet for point cloud recognition," in *CVPR*, 2018, pp. 4606–4615.

[21] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *CVPR*, 2018, pp. 7794–7803.

[22] Y. Lyu, X. Huang, and Z. Zhang, "Learning to segment 3d point clouds in 2d image space," in *CVPR*, 2020, pp. 12 255–12 264.

[23] S. Qiu, S. Anwar, and N. Barnes, "Dense-resolution network for point cloud classification and segmentation," in *WACV*, 2021, pp. 3813–3822.

[24] L. Han, T. Zheng, L. Xu, and L. Fang, "Occuseg: Occupancy-aware 3d instance segmentation," in *CVPR*, 2020, pp. 2940–2949.

[25] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, and J. Jia, "Pointgroup: Dual-set point grouping for 3d instance segmentation," in *CVPR*, 2020, pp. 4867–4876.

[26] C. R. Qi, X. Chen, O. Litany, and L. J. Guibas, "Imvotenet: Boosting 3d object detection in point clouds with image votes," in *CVPR*, 2020, pp. 4404–4413.

[27] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang, "Mlcvnet: Multi-level context votenet for 3d object detection," in *CVPR*, 2020, pp. 10 447–10 456.

[28] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *CVPR*, 2019.

[29] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz, "Splatnet: Sparse lattice networks for point cloud processing," in *CVPR*, 2018, pp. 2530–2539.

[30] F. Engelmann, T. Kontogianni, and B. Leibe, "Dilated point convolutions: On the receptive field size of point convolutions on 3d point clouds," in *ICRA*, 2020, pp. 9463–9469.

[31] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *ECCV*, 2018, pp. 87–102.

[32] 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 *ICCV*, 2019, pp. 6411–6420.

[33] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *CVPR*, 2018, pp. 7132–7141.

[34] M. Feng, L. Zhang, X. Lin, S. Z. Gilani, and A. Mian, "Point attention network for semantic segmentation of 3d point clouds," *Pattern Recognition*, vol. 107, p. 107446, 2020.

[35] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *CVPR*, 2019, pp. 10 296–10 305.

[36] C. Chen, L. Z. Fragonara, and A. Tsourdos, "Gapointnet: Graph attention based point neural network for exploiting local feature of point cloud," *Neurocomputing*, vol. 438, pp. 122–132, 2021.

[37] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv:1710.10903*, 2017.

[38] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point transformer," *arXiv:2012.09164*, 2020.

[39] M.-H. Guo, J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu, "Pct: Point cloud transformer," *Computational Visual Media*, vol. 7, p. pages187–199, 2021.

[40] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *arXiv:2101.01169*, 2021.

[41] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[42] J.-H. Kim, K.-W. On, W. Lim, J. Kim, J.-W. Ha, and B.-T. Zhang, "Hadamard product for low-rank bilinear pooling," 2017.

[43] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *CVPR*, 2010, pp. 2559–2566.

[44] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv:1312.4400*, 2013.

[45] D. Misra, "Mish: A self regularized non-monotonic neural activation function," *arXiv:1908.08681*, 2019.

[46] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *CVPR*, 2015, pp. 1912–1920.

[47] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *ICCV*, 2019.

[48] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "Pointcnn: Convolution on x-transformed points," in *NIPS*, 2018, pp. 820–830.

[49] M. Xu, R. Ding, H. Zhao, and X. Qi, "Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds," in *CVPR*, 2021, pp. 3173–3182.

[50] H. Zhou, Y. Feng, M. Fang, M. Wei, J. Qin, and T. Lu, "Adaptive graph convolution for point cloud analysis," in *ICCV*, 2021, pp. 4965–4974.

[51] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *CVPR*, 2018, pp. 4558–4567.

[52] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "Pointweb: Enhancing local neighborhood features for point cloud processing," in *CVPR*, 2019, pp. 5565–5573.

[53] Z. Hu, M. Zhen, X. Bai, H. Fu, and C.-l. Tai, "Jsenet: Joint semantic segmentation and edge detection network for 3d point clouds," *arXiv:2007.06888*, 2020.

[54] S. Fan, Q. Dong, F. Zhu, Y. Lv, P. Ye, and F.-Y. Wang, "Scf-net: Learning spatial contextual features for large-scale point cloud segmentation," in *CVPR*, 2021, pp. 14 504–14 513.

[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[56] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *CVPR*, 2015, pp. 567–576.

[57] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," in *CVPR*, 2016, pp. 808–816.

[58] Z. Ren and E. B. Sudderth, "Three-dimensional object detection and layout prediction using clouds of oriented gradients," in *CVPR*, 2016, pp. 1525–1533.

[59] J. Lahoud and B. Ghanem, "2d-driven 3d object detection in rgb-d images," in *ICCV*, 2017, pp. 4622–4630.

[60] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *CVPR*, 2018, pp. 918–927.

[61] Q. Xie, Y.-K. Lai, J. Wu, Z. Wang, Y. Zhang, K. Xu, and J. Wang, "Vote-based 3d object detection with context modeling and sob-3dnms," *IJCV*, 2021.