Pyramid Architecture for Multi-Scale Processing in Point Cloud Segmentation

Dong Nie, Rui Lan, Ling Wang, Xiaofeng Ren
AMap, Alibaba
{dong.nie, lr264907, wangling.lingwang, x.ren}@alibaba-inc.com

Abstract

Semantic segmentation of point cloud data is a critical task for autonomous driving and other applications. Recent advances of point cloud segmentation are mainly driven by new designs of local aggregation operators and point sampling methods. Unlike image segmentation, few efforts have been made to understand the fundamental issue of scale and how scales should interact and be fused. In this work, we investigate how to efficiently and effectively integrate features at varying scales and varying stages in a point cloud segmentation network. In particular, we open up the commonly used encoder-decoder architecture, and design scale pyramid architectures that allow information to flow more freely and systematically, both laterally and upward/downward in scale. Moreover, a cross-scale attention feature learning block has been designed to enhance the multi-scale feature fusion which occurs everywhere in the network. Such a design of multi-scale processing and fusion gains large improvements in accuracy without adding much additional computation. When built on top of the popular KPConv network, we see consistent improvements on a wide range of datasets, including achieving state-of-the-art performance on NPM3D and S3DIS. Moreover, the pyramid architecture is generic and can be applied to other network designs: we show an example of similar improvements over RandLANet.

1. Introduction

With the rise of autonomous driving, semantic segmentation of point cloud data is increasingly drawing attention in research. Building deep models for point clouds, sets of orderless points at arbitrary 3D positions, is arguably different from that for images. Early works projected 3D points to regular structures so that convolution operators could be used [31, 36, 40]. Later, the pioneering work of PointNet [33,34] developed a promising method to directly apply deep learning on sparse 3D points, using shared multi-layer perceptrons (MLPs) to learn per-point features.

Follow-up work along the line of PointNet typically consists of three key components, namely: point-wise transformation, local aggregation, and point sampling. Local aggregation operator plays a similar role for points as the convolution layer does for image pixels [27]; and point sampling works as a pooling layer does for pixels [34, 52, 60]. To take the similarities further, state-of-the-art point cloud segmentation methods mostly employ the encoder-decoder U-shape architecture [13, 34, 47], which is a classic design in image segmentation (UNet [37]). In the encoder path, transformation layers learn increasingly sophisticated per-point features, local aggregation operators combine information in local neighborhoods, and point subsampling layers further increase the receptive field. The decoder path consists of upsampling and per-point transformation layers.

Most recent works on point cloud segmentation focused on either local aggregation [14, 19, 20, 27, 30, 34, 45, 47, 53, 58] or point sampling strategies [1, 8, 13, 24, 41, 54, 55]. For example, PointNet++ [34] applied several MLPs on a concatenation of relative position and point feature to aggregate information in local neighborhoods. KPConv [47] designed to obtain pseudo grid feature and applied convolution on these kernel points. RandLANet [13] compared point sampling methods and selected random sampling for efficiency. Density-adaptive sampling [1] was proposed to handle heterogeneous density distributions and class imbalance.

Interestingly, for point cloud segmentation, little attention has been devoted to the study of the network architecture itself. This is in stark contrast with image segmentation, where most recent efforts went way beyond the basic encoder-decoder U-structure to design better and more efficient architectures, especially on the topics of multi-scale processing and fusion [18, 32, 35, 48, 57] and context aggregation [5, 28, 50, 59]. For example, HRNet [48] proposed to aggregates multi-scale features throughout lateral stages, with an emphasis on high-resolution representation. Hierarchical Attention [44] was also built upon better uses of multi-scale information, from a perspective of inference.

Point cloud data, no different from images, are multi-

Code is available at https://github.com/ginobilinie/kp_pyramid scale in nature and requires multi-scale processing, including the need to balance large-scale context with fine detail, and the potential use of multiple local aggregation stages in order to extract semantic information. In this work, we show that indeed there is an urgent need, and a substantial benefit, to move beyond the U-shape structure in point cloud segmentation. Inspired by latest advances in image segmentation [32, 48], we open up the standard encoder-decoder architecture to design a pyramid architecture for point cloud segmentation (see Fig 1). A number of design improvements are proposed and validated:

- we use lateral stages to link up the counterparts in the encoder and decoder paths at each scale, where neighborhoods are re-used in local aggregation and sampling;
- we add upward/downward links to form a full "pyramid" shape which allows information at varying scales and stages to be fused;
- we identify three components in fusion, design a novel Cross-scaLe Attention fusIon Module (CLAIM, which is almost parameter-free) to better serve the aggregation of multi-scale features, and empirically find the best choices.

Note each of these is novel for point cloud segmentation, and together they provide a substantial boost in accuracy without a higher demand on computation. When built up on the popular KPConv network as the baseline, our pyramid architecture leads to 1.0 $\sim$ 3.0% improvements in mIoU on a wide range of benchmarks for both outdoor and indoor scenes, including SOTA results on NPM3D and S3DIS (with mIoU of 83.0 and 73.0, respectively). Moreover, our pyramid architecture is generic and can be used to enhance any encoder-decoder network. For example, when using the more efficient RandLANet [13] as the baseline, similar large improvements in accuracy are also observed.

2. Related Works

Point-based 3D Segmentation Networks The pioneering work PointNet [33] is proposed to directly handle point cloud analysis which learns per-point features using shared MLPs and global features using symmetrical pooling functions. Inspired by PointNet, a series of point-based networks have been designed. These methods could be generally categorized into four types: (a) point-wise MLP based, (b) pseudo grid features based, (c) Recurrent Neural Networks based (RNN-based) and (d) graph-based methods. (a) The point-wise MLP based methods usually use shared MLP as the basic unit in their network [33]. Though being quite efficient, point-wise features extracted by shared MLP cannot capture the local geometry in point clouds. PointNet++ [34] has a neighborhood grouping module to capture wider context for each point and learn richer local structures. Hu [13] proposed local feature aggregation module to enlarge the receptive field so that the followed random sampling may not miss much information. Attention (weighted-sum) based local aggregation methods have also been largely studied [6,27,54]. (b) Among pseudo grid feature based methods, KPConv [47] is a representative work, in which, predefined number of equally distributed spherical grid points are sampled and the pseudo features for a pseudo grid point are computed based on distance from the real points within the sphere. The kernel weights can then be easily learned since the number of pseudo points are fixed during training. There are some other pseudo grid feature based methods [14, 20, 30, 45, 58], the key difference lies in the definition of the pseudo points. (c) RNN-based methods target at capturing inherent context features from point clouds [9, 15, 56] with the advance of recurrent module. (d) Graph-based methods aims at learning the underlying shapes and geometric structures of 3D point clouds [21,22,49,61].

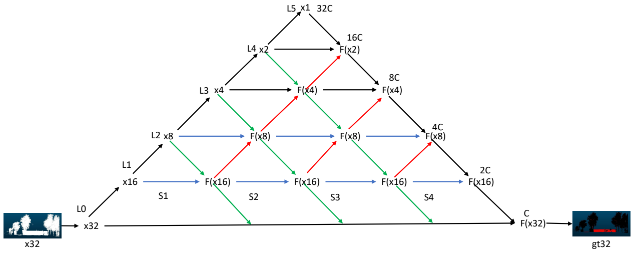

Figure 1. Illustration of the proposed pyramid architecture. The pyramid network has lateral paths containing up to four stages. Each stage involves a tridirectional fusion module to boost information integration between scales. Lateral flow (blue) applies transformation while maintaining resolution; top-down flow (green) provides context for higher resolution paths; and bottom-up flow (red) down-samples high-resolution features to assist in extracting semantics at higher levels. The F is the designed multi-scale feature fusion block shown in Fig. 2.

Among the above-mentioned point-based 3D segmentation networks, the mostly adopted network architecture is U-shape encoder-decoder network [13, 27, 34, 47], which demonstrate the success and popularity of the U-shape based network for segmentation.

Multi-Scale Semantic Segmentation Networks In the U-shape encoder-decoder networks [37], an encoder usually reduces the spatial resolution of feature maps to learn more abstract features. Correspondingly, the decoder recovers the spatial resolution of the input image from encoder so as to generate dense prediction maps. The skip connection combines shallow and deep features with skip connections to retain more details in the dense predictions. Many works have been done in semantic segmentation to utilize the multi-scale information to achieve robustness and higher accuracy. PANet [26] built a bottom-up connection between lower layers and the topmost layer to enhance the encoder-decoder's feature hierarchy with better localization and small-scale detail in the lower layers. HRNet [42] introduced multi-resolution convolution to fully fuse multi scale information, and the high-resolution pathway can well retain the localization information. BPNet [32] proposed a pyramid network with top-down and bottom-up information flow, to enhance information interaction between large-scale contexts and small-scale details. In a related work on object detection, EfficientDet [43] proposed a weighted bi-directional feature pyramid network (BiFPN), showing that information flow in both directions (coarse-to-fine, and fine-to-coarse) are useful for feature fusion. As for point cloud segmentation, only a few works walk towards using multi-scale information. PointNet++ [34] adaptively combines features from multiple scales in a hierarchical manner. [29] fused both global and local features in multiple scales to endow the segmentation network with more discriminative features. PointSIFT [17] consider multi-scale information to form a robust feature extractor.

3. Approach

The architecture design of our Pyramid structure to process multi-scale information is illustrated in Fig 1, including a preliminary feature transformation step, a pyramid network for cross-scale information processing and fusion, followed by feature transformation layers to generate the final dense prediction for segmentation.

3.1. Tridirectional Pyramid Architecture

As shown in Fig. 1, in addition to the encoder-decoder path (black), the proposed pyramid architecture has lateral links (in multiple stages) in the horizontal direction for all layers (levels) except the bottom layer (*i.e.*, $L0$ (we do not introduce lateral links in $L0$ to avoid large computational cost). Each stage in the pyramid involves a local aggregation operation, so that the lateral (horizontal) information flow can have increasingly large receptive fields but also keep the spatial resolution without losing detail. Moreover, at each stage, we add links in the vertical directions to boost cross-scale interaction.

Information (and processing) in the pyramid can flow in three directions, illustrated in different colors: one moves "forward" (laterally, in blue color) in stages, maintaining spatial resolution while applying local aggregation operators to integrate information; one moves "down' (in green) in layers, adding larger-scale context to finer-scale detail; the third moves "up" (in red) in layers, from higher spatial resolution to lower resolution, providing richer information for larger contexts. The original encoder-decoder paths are shown in black in the figure. A "bird-eye" view of our network resembles a pyramid, or a Pascal Triangle.

3.1.1 Lateral Information Flow

A typical instantiation of our pyramid structure consists of 4 or 5 layers (3 or 4 steps of subsampling). If the input resolution is x32 (with x an arbitrary integer), the feature resolution of the lowest layer is x32. At the second lowest layer (*i.e.*, the 1st layer in Fig.1), the feature resolution is x16, and it goes through 3 lateral stages of local aggregation and transformation to learn better features and to enlarge the effective receptive field. Because the spatial resolution (*i.e.*, number of points) remains the same, such a lateral link can thus learn high-resolution representation, especially in the low layers (*e.g.*, L1). Note we donot use local aggregations in the lateral links from deep layers (*e.g.*, L3 and L4) since we cannot gain much more semantics with such operations there. As we move up the "pyramid", following common practice, we reduce the feature resolution by half at each step, and increase the number of channels by two (channel numbers are shown as C, $2C$, $4C$, etc. with C an integer).

Note that we do not simply add direct lateral links, as skip connections often do. For each lateral link, we have a varying number of stages. Typically, there are 3 or 4 stages of local aggregation at the 1st layer. As we move up the layers, fewer processing steps are needed laterally, as the incoming information already passes through a number of local aggregation in the subsampling process.

By adding the lateral links with varying stages, a pyramid structure takes shape. It is substantially different from the original U-Shape structure with a single encoder-decoder path (or, for that matter, a typical feature pyramid for object detection). There is no clearly defined encoder or decoder. This structure allows us to further add links to enable cross-scale information fusion (see below).

3.1.2 Cross-scale Information Flows

Top-down information flow. With aforementioned lateral-link based architecture, we describe how we design cross-scale information flow in a systematic way. One component is *top-down* information flow. As shown in Fig 1 (in green arrows), information flows "down" the pyramid at each processing step. For example, the features at L2 (at resolution x8 and after one step of local aggregation with subsampling) are fed down the hierarchy to be integrated with S1, which is one step of local aggregation at the 1st layer (maintaining feature resolution x32). Similarly, the features at L3 (at resolution x4) are fed down the hierarchy to the layer below, to be integrated with the output of one lateral step from L2. Other top-down flows are designed similarly across the pyramid structure, at all layers and all stages.

Bottom-up information flow. The top-down flows in our pyramid network enhances processing at lower layers (higher resolution) with more contextual and semantic information from higher layers. However, cross-scale information flow does not have to be in only one direction. We also add bottom-up information flows, as illustrated by red arrows in Fig. 1. For bottom-up flows, higher-resolution features (after top-down fusion) are fed upward to be integrated with lower-resolution features at higher layers. This design completes our pyramid network for multi-scale processing: information is free to flow laterally, upward, or downward, and they are fused at every step of the processing. In the ablation studies, we will show that all three types of flows (lateral, top-down and bottom-up) are useful and provide substantial improvements in accuracy.

Empirically, the number of layers (in the resolution hierarchy) and the number of stages (processing steps at the lowest layer) tend to be the same, which results in a "perfect" triangle. In all the models we use, the triangles are "perfect", and they produce good results across board. Meanwhile, the number of layers and the number of stages do not have to be the same. We have experimented with "skewed" triangles and they can be effective under certain circumstances (such as when the input resolution is high but we want a lighter weight model).

3.1.3 Multi-Scale Feature Fusion Strategy

The tridirectional information flows in our pyramid network bring together features at different scales with different characteristics. It is natural that how we fuse these features plays a central role in the design.

In image segmentation, people often use element-wise *addition*, element-wise *multiplication*, or *concatenation*, together with conv1x1 and conv3x3 to formulate an entire feature fusion process [48]. In recent point cloud segmentation works, *concatenation* is typically used and followed by one or several MLPs to do feature transformation [27]. We carry out a systematic design and its empirical validation.

General Formulation. A multi-scale fusion module consists of three components: pre-fusion transformation (including scale matching), fusion, and post-fusion transformation. In general, for K input vectors at different scales, $s_1, s_2, ..., s_K$, we select a base scale s_m and fuse feature vectors at other scales to this base scale m. Upsampling or downsampling is adopted to match the base scale and align the spatial dimensions. Transformations G (*e.g.*, MLPs or local aggregators (LA) or identity mapping (IM)) may be applied to each of these scales, a fusing operator F (typical using concatenation (CAT), element-wise sum (SUM), element-wise multiplication (MUL), element-wise weighted sum (wSUM), element-wise max-pooling(MAX)) is then utilized to aggregate all transformed features, and finally the output feature vector g can be obtained after a transformation function (T) (*e.g.*, MLPs or LA or IM).

$$g = T(F(\{G_i(s_i)|i = 1, 2, ...K\})) \qquad (1)$$

The three-component design of fusion is shown in Eq. 1. We also illustrate the design in Fig. 2, in which, the base scale is B. For most of the fusion modules in our network, there are three inputs from three scales, with the middle scale being the base.

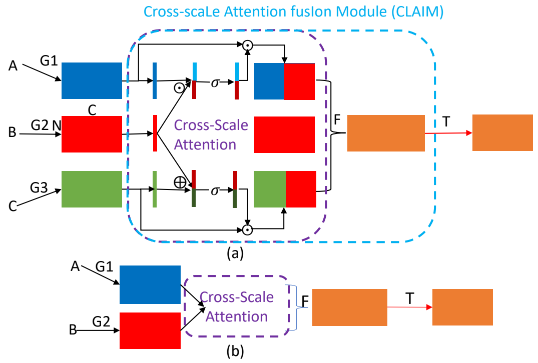

Figure 2. Illustration of our cross-scale attention based multi-scale feature fusion block, consisting of pre-fusion transformation, cross-scale attention feature learning, feature fusion, and post-fusion transformation (see text). (a) denotes 3-scale fusion for the levels 2 and above. (b) denotes 2-scale fusion for level 1.

Cross-scaLe Attention fusIon Module (CLAIM). Since the pre-fusion transformation and post-fusion transformation are usually standard operations (*e.g.*, in our work, $G1$ and $G3$ are identity mapping, $G2$ can be LA, and T can be MLP), we focus on designing the multi-scale fusion module. In our work, we carefully design a cross-scale attention based feature learning block, which fully considers the characteristics of features from different scales to boost learning high-resolution semantic features, to work as the core to enhance the fusion module. Besides the base scale B, A is from the higher level (low-res, representing context), B is from the base level and C from the lower level (high-res, representing detail). Compared to B, A has richer context information, and C has more details. They have different characteristics and the ideal fusion is to retain the details from C and keep the semantics from A. The proposed cross-scale attention based feature learning block is designed to enhance detail features in C and semantic features in A by interacting the neighbor-scale features.

Note that directly element-wise summing B and C can bring detail information but would tend to produce blurred boundaries since context information in B has a low resolution, and intuitively multiplying A and B element-wise allows information both in A and B to reinforce each other, but unique signals in either A or B could be suppressed. Instead, we first squeeze the channels of the features in all the three scales (A, B and C) to 1. Then we conduct a "ADD" operation on B and C, a "MUL" operation on B and A, to obtain the spatialwise attention masks (for convenience, we name them sem-mask (M_{sem}) and res-mask (M_{res}) respectively) with a sigmoid activation, as shown in Eq. 2 and Eq. 3.

$$M_{sem} = \sigma\left(z\left(A\right) \cdot z\left(B\right)\right) \tag{2}$$

$$M_{res} = \sigma\left(z\left(C\right) + z\left(B\right)\right) \tag{3}$$

Following the above-mentioned steps, we apply the M_{sem} on A and M_{res} on C so that their own characteristics are enhanced without suffering the shortcomings, as shown in Eq. 4 and Eq. 6. As mentioned before, we also apply local aggregation on B (Eq.5) to achieve even higher semantics (blue link in Fig. 1.

$$A' = A \odot M_{sem} \tag{4}$$

$$B' = g(B) \tag{5}$$

$$C' = C \odot M_{res} \tag{6}$$

With characteristic-enhanced multi-scale features, we stack them (*i.e*., A', B' and C') together (F in Fig. 2) to aggregate multi-scale features. Then we employ a MLP (T in Fig. 2) to reduce the channels of the stacked features. The settings for 2-scale feature fusion is similar with 3-scale feature fusion. For the entire fusion module, we name it Cross-scaLe Attention fusIon Module (CLAIM).

It is worth-noting that we design the cross-scale attention mechanism in a (almost) parameter-free manner instead of more complicated ones, because we would like to avoid introducing much more parameters (point cloud segmentation networks can be easily overfitting). Our strategy is proved to be effective in ablation study.

Reuse of Local Neighborhoods. Each of our fusion step involves local aggregation operations. Typically, local neighborhoods are computed using either a radius query or a KNN query based on distance, both of which can be computationally expensive. Fortunately, we can re-use such local neighborhoods: for each lateral link (and downsampling), regardless of the number of stages, only one neighborhood query is needed. This allows our pyramid architecture to be efficient and does not incur a large increase in computation comparing to the baseline.

We will demonstrate in ablation studies that CLAIM is empirically the best choice and performs better than other combinations. In addition, CLAIM does not need more parameters than other fusion blocks.

3.2. Making KP-Pyramid and RandLA-Pyramid

The proposed pyramid architecture can apply to any encode-decoder based segmentation network. We take KPConv as an example to show how we 'upgrade' KPConv to KP-Pyramid. Shown in Fig. 1, we adopt 'KPConv' operators in the blue arrows to work as the local aggregation operators, and shared MLPs (unary convolution) in the red arrows. For downsampling and upsampling, we follow the KPConv settings to use strided KPConv (we can also use max-pooling) and nearest point upsampling. We use the designed CLAIM shown in Fig. 2 to complete the multo-scale feature fusion within the pyramid. For the other settings, for example, the number of channels and rigid or deformable kernel, we directly follow the ones in KPConv [47]. To this end, we have successfully 'upgraded' KPConv to KP-Pyramid.

As an example of flexibility, we also adapt the more efficient RandLANet [13] to the pyramid architecture. We adopt the local feature aggregation (LFA) module to learn neighborhood features in the upward links. We adopt a simplified LFA, which removes the dilated residual block, for lateral feature transformation. MLPs are used for the other links. Random sampling is used for downsampling, and nearest-neighbor interpolation is used for the point feature upsampling. Note the downsampling/upsampling ratio between layers is not set to 2 as shown in Fig. 1, but follows the settings in RandLANet. In this way, we convert RandLANet to RandLA-Pyramid.

Table 1. Comparing pyramid architecture (KP-Pyramid) with U-shape architecture (original KPConv), using the standard mIoU metric. As can be clearly seen, the pyramid architecture provides a substantial improvement in accuracy, consistently aross all three datasets, and for both rigid and deformable settings of KPConv.

Methods	PL3D	S3DIS	Semantic3D
KPConv rigid	77.8	69.1	74.6
KP-Pyramid rigid	80.5	71.7	76.4
ΔmIoU	+2.7	+2.6	+1.8
KPConv deform	81.2	70.6	73.1
KP-Pyramid deform	83.0	73.0	75.8
ΔmIoU	+1.8	+2.4	+2.7

4. Experiments and Results

4.1. Datasets and Settings

We carry out experimental validations of our pyramid architecture on three commonly used point cloud benchmarks, including a variety of indoor and outdoor scenes: (1) Paris-Lille-3D (PL3D) [39], a segmentation challenge of NPM3D, for outdoor mobile scans; (2) S3DIS [2], for indoor large spaces and (3) Semantic3D [11], for outdoor fixed scans. PL3D contains more than 2km of streets in 4 different cities and is an online benchmark. The 160 million points of this dataset are annotated with 10 semantic classes, and 30 million points collected in three cities works as test set. S3DIS covers six large-scale indoor areas from 3 buildings for a total of 273 million points labeled with 13 classes. For S3DIS, we follow experimental protocols in [13, 47] and use k-fold and Area-5 as test scene to measure the generalization ability of our method. Semantic3D is an online benchmark with several fixed lidar scans of dif-

Table 2. Investigation of tridirectional information flow. All three types of information flow (link) provide boost in performance. Results are computed under the same setting.

method	fusion strategy	mIoU	ΔmIoU
BaseNet	-	66.0	-
+lateral	-	66.6	+0.6
+lateral+downward	CLAIM	67.6	+1.6
+lateral+upward	CLAIM	67.2	+1.2
+pyramidal	CLAIM	68.2	+2.2

ferent outdoor scenes, and it has more than 4 billion points with 8 semantic categories. We again follow experimental protocols in [13, 47] and select the reduced-8 challenge because it is less biased by the objects close to the scanner.

We use the official open source code of KPConv as the baseline and build on top of it. KPConv is a state-of-the-art method for point cloud segmentation and has been widely used. For training settings, We use the hyper-parameter settings in KPConv [47] as provided by the open source code on NPM3D, S3DIS and Semantic3D since there is not official KPConv experimental settings). For example, we set K = 15, $\Sigma = 1.0$ and $\rho = 5.0$ for all experiments. The setting of the convolution radius is also exactly the same as those in KPConv on all the three datasets. Also, the first subsampling cell size dl_0 is determined by the dataset and $dl_{j+1} = 2 \times dl_j$.

4.2. Improvements over the U-Shape Baseline

First, we show experimental results on all three datasets, comparing our pyramid architecture (KP-Pyramid, the pyramid version of KPConv) with the U-Shape encoder-decoder baseline (the standard KPConv). In this comparison, to make it fair, we use results from the KPConv open source code with provided settings (evaluated on online servers when needed). Note that the results from the open source code may be different from those in the paper or in online benchmarks, sometimes higher, other times lower.

The experimental results are presented in Table 1. The settings are mostly kept consistent between the baseline and the pyramid-enhanced network. PL3D and Semantic3D scores are obtained on test datasets. S3DIS scores are obtained using k-fold cross-validation. Endowed with the pyramid structure to process and fuse multi-scale information, the performance on all datasets are improved. On NPM3D, the pyramid structure provides a performance gain of more than 2.2 mIoU points in average; On S3DIS, the performance gain is up to 2.7 points and on Semantic3D, the average gain is more than 2.0 points. We thus show that the proposed pyramid architecture, with better multi-scale processing and fusion, significantly improves the baseline.

4.3. Ablation Studies

Our pyramid architecture has a number of novelties over the baseline, including the lateral information flow, the cross-scale upward and downward information flows, and the choice of the fusion strategy. How much do they help? How do they compare to alternative choices? We conduct ablation studies to answer these questions. The experiments are conducted on S3DIS, using area 5 set for evaluation.

4.3.1 Impact of Pyramidal Information Flows

To investigate the effects of the added links (information flows) within the pyramid architecture, we compare the following networks: (a) BaseNet which is the same architecture as KPConv Deformable; (b) adding lateral links in the intermediate layers of the BaseNet as shown in Fig. 1 which represents the 'lateral' or 'forward' information flow (denoted as '+lateral'); (c) adding only top-down flow in the pyramid network, which is the downward information flow (denoted as '+lateral+downward'); (d) adding only bottom-up flow, which indicates the upward information flow (denoted as '+lateral+upward'), and (e) adding both top-down and bottom-up flows, which completes the pyramid shape with the lateral flows (denoted as '+pyramidal'). The results are shown in Table 2.

As shown in the table, '+lateral' provides an improvement of 0.6 mIoU point, showing modest gains by adding a direct link for each scale with more 'convolution' stages. On top of the network with lateral links, both downward and upward information flow can further boost the network to achieve better performance. Compared to upward information flow, downward information flow is more beneficial, which confirms that providing context to high-resolution processing is more important. With both the downward and upward links, the network can enjoy an even larger performance gain, demonstrating the merit of having information flow at every step of the processing, in all forward (lateral), upward and downward directions.

4.3.2 Impact of Multi-Scale Fusion Strategies

Described in Sec. 3.1.3, we formulate the multi-scale information fusion to be combinations of transformation layers and fusing operators, as shown in Eq. 1. To validate the effectiveness of our designed CLAIM and also explore which factors play important roles in multi-scale feature fusion, we conduct comprehensive experiments to understand the impact of different multi-scale fusion strategies. In particular, we use the proposed bidirectional pyramid architecture as the basis, and compare different choices of pre-fusion transformation (*i.e.*, IM and MLP), fusion and post-fusion transformation (*i.e.*, IM and MLP). Especially, we take the following choices for the fusion strategy, namely, direct fusion (*i.e.*, CAT, SUM, MAX and MUL) and attention based fusion which are listed below:

- S3SE: Stack A, B and C, then apply SE module (channel attention) [12] to enhance them.
- S2SES: Stack A, B, apply SE to enhance A, and do the similar to C. Then we stack all them.
- S3CBAM: Stack A, B and C, then apply CBAM (dual channel and spatial attention) [51] to enhance them.

The results are presented in Table 3. These results validate that our choice of 'IM+CLAIM+MLP' provides the highest score. They also provide other insights into fusion. For pre-fusion transformation, it is also interesting to see that IM performs better than MLP when followed by a suitable fusion (*e.g.*, CAT or CLAIM), which suggests that having more sophisticated modules may not help here, as there is a risk of overfitting. For feature fusion, our proposed CLAIM are is a good choice and CAT is an alternative for direct fusion. It is worth-noting that the widely used attention blocks [7,12,51] for multi-scale feature fusion in image recognition cannot directly work well in point cloud segmentation cases since they can be much more easily overfitting, which means one key point to the success of CLAIM is its (almost) parameter-free design. The benefit of CLAIM can also be attributed to that scale A contains rich semantic information, in other words, good representation for large objects, and C contains more local details which is good for tiny objects and boundaries. For post-fusion transformation, using LA as transformation after feature aggregation does not seem a good choice; channel-wise transformation is more effective. This is consistent with practice in image segmentation, where conv1x1 is commonly used after feature fusion.

Table 3. Investigation of multi-scale feature fusion strategies. 'Pre-Fusion-T' represents feature transformations for the incoming scales before fusion (note $G2$ is fixed as LA). 'Fusion' is the feature fusion operator. 'Post-Fusion-T' denotes transformation after fusion.

Pre-Fusion T ΔmIoU	Fusion	post-Fusion T	mIoU
MLP	CAT	MLP	64.7
MLP	SUM	MLP	66.8
MLP	MAX	MLP	67.1
MLP	SUM	IM	66.2
MLP	MAX	IM	66.7
MLP	MUL	IM	63.8
IM	**CAT**	**MLP**	**67.9**
IM	CAT	LA	66.6
IM	S3SE	MLP	66.2
IM	S2SES	MLP	66.4
IM	S3CABM	MLP	65.2
IM	**CLAIM**	**MLP**	**68.2**

4.3.3 Efficiency Analysis of the Pyramid Architecture

In this section, we evaluate the overall efficiency of the proposed pyramid structure on real-world datasets for semantic segmentation. In particular, we measure running time and memory cost of KP-Pyramid on NPM3D and Semantic3D test set, where NPM3D test set contains 3 areas, each with 10 million points and Semantic3D test set contain 4 areas, each with 10 to 30 million points. For a fair comparison, we set the same *num_votes* (set to 4) for all the networks during inference. We keep the test configuration identical for the different methods on each dataset (*e.g.*, conv_radius, batch_num etc.). The inference is carried out on a Nvidia RTX 2080 TI card with torch 1.5.1.

As shown in Table 5, with pyramid structure for multi-scale information process and fusion, the increase of inference time is around 11%. The memory increases by around 15% at inference time. The experimental results validate the efficiency of our proposed pyramid structure, even though we add several multi-scale processing and fusion operations inside the pyramid. As discussed, one key to efficiency is the re-use of neighborhood radius queries.

4.4. Comparison with State-of-the-Art Results

Previously we have evaluated our proposed pyramid architecture against the baseline encoder-decoder U-shape architecture, and conducted ablation studies to validate its components. To have a more comprehensive view of how our network compares to the state of the art, we report detailed results alongside other recent methods. The listed results are either from the published papers or online benchmark evaluations (when available). We follow the settings in KPConv without change for Paris-Lille-3D, S3DIS and Semantic3D.

The results are shown in Table 6. Our model outperforms all existing methods on NPM3D and S3DIS by considerable margins. In particular, we achieve mIoU 83.0 on NPM3D, and 73.0 on S3DIS, both setting new records on popular benchmarks. We also improve the KPConv on Semantic3D to a mIoU of 76.4.

Class-wise details of the results are provided for S3DIS in Table 4. Our results tend not to have 'weak spots', i.e. there are no classes that have very low accuracy. Furthermore, we do quite well on both small objects (*e.g.*, book) and large objects (*e.g.*, ceil, floor), despite it being an indoor setting vs an outdoor setting. This is a testament to the merit of a multi-scale fusion architecture. Good performance on small objects can be attributed to the high-resolution feature representation in the first layer in Fig. 1; for the performance on large objects, the design of our pyramid architec-

75.9 is reported in the original KPConv paper. The latest mIoU for KPConv in the evaluation benchmark is 82.0. However, we donot have the experimental setting for this result from the github source code.

Table 4. Semantic segmentation IoU scores on S3DIS k-fold. Additionally, we give the mean class recall, a measure that some previous works call mean class accuracy.

Methods	mIoU	mRec	ceil.	floor	wall	beam	col.	wind.	door	chair	table	book.	sofa	board	clut.
Pointnet [33]	47.6	66.2	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 [16]	56.5	66.5	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
SPGraph [23]	62.1	73.0	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
PointCNN [25]	65.4	75.6	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
RandLANet [13]	70.0	82.0	93.1	96.1	80.6	62.4	48.0	64.4	69.4	69.4	76.4	60.0	64.2	65.9	60.1
SCFNet [10]	71.6	82.0	93.3	96.4	80.9	64.9	47.4	64.5	70.1	71.4	81.6	67.2	64.4	67.5	60.9
KPConv rigid [47]	69.6	78.1	93.7	92.0	82.5	62.5	49.5	65.7	77.3	57.8	64.0	68.8	71.7	60.1	59.6
KPConv deform [47]	70.6	79.1	93.6	92.4	83.1	63.9	54.3	66.1	76.6	57.8	64.0	69.3	74.9	61.3	60.3
KP-Pyramid deform	73.0	82.2	94.6	95.5	84.1	63.0	56.8	70.9	78.6	67.8	69.2	67.5	78.3	58.4	64.4

Table 5. The computation time (seconds) and memory cost (GB) for inference on test sets of PL3D (NPM3D) and Semantic3D datasets. The additional cost of using our pyramid architecture is minimal.

Method	NPM3D		Semantic3D	
	Memo	Time	Memo	Time
KPConv deformable	3-4.2	193	5.5-7.5	274
KP-Pyramid deformable	3.4-4.7	216	5.9-8.2	290

Table 6. 3D scene segmentation scores (mIoU). PL3D (NPM3D), Semantic3D scores are taken from their respective online benchmarks (reduced-8 challenge). S3DIS scores are given by k-fold cross validation.

Methods	PL3D	S3DIS	Semantic3D
RF_MSSF [46]	56.3	49.8	62.7
MSDVN [38]	66.9	54.7	65.3
SPGraph [23]	-	58.0	73.2
ConvPoint [3]	75.9	68.2	76.5
SCFNet [10]	-	71.6	77.6
KFAConv [4]	82.7	68.4	74.6
RandLANet [13]	78.5	70.0	77.4
KPConv rigid [47]	72.3	69.6	74.6
KPConv deform [47]	75.9(82.0)	70.6	73.1
RandLA-pyramid	80.1	71.5	77.5
KP-Pyramid rigid	80.5	71.7	76.4
KP-Pyramid deform	83.0	73.0	75.8

ture allows richer information to flow 'upward' at various stages, not just on a single encoder path.

4.5. Additional Experiments based on RandLANet

The above-introduced experiments and results indicate the success of our adaptation for KPConv [47], that is, improving multi-scale processing and fusion in the encoder-decoder segmentation architectures. To investigate the generalization ability of our proposed multi-scale processing and fusion strategy, we conduct additional explorations on another typical encoder-decoder based point cloud segmentation network, RandLANet [13]. The 'upgrade to RandLA-Pyramid' process is introduced in Sec. 3.2.

We test RandLA-Pyramid on several datasets, with the results shown in Table 6. On NPM3D and S3DIS, RandLA-Pyramid achieves about 1.5 points improvement in terms of mIoU against the baseline. At the same time, the inference time and memory cost do not increase much (*i.e.*, less than 10%). This demonstrates that our proposed pyramid structure is generic and can potentially apply to any encoder-decoder networks.

5. Conclusion

We presented a tridirectional pyramid architecture to process and fuse multi-scale information for point cloud segmentation. We improved the commonly used encoder-decoder structure with several simple and yet effective components, *i.e.*, lateral as well as top-down and bottom-up information flows and a scale pyramid architecture, to enhance interaction between large-scale contexts and small-scale details. We also explored feature fusion strategies for cross-scale feature fusion within the pyramid structure and designed the effective (almost) parameter-free CLAIM for multi-scale feature fusion. State-of-the-Art results were obtained on standard benchmarks and the proposed components were shown to provide substantial improvements in accuracy. Without needing pre-training, we believe our model have the potential to be used for many point-cloud related applications and still have room for further improvements.

6. Social Impact and Limitations

Our proposed point cloud algorithm can boost the development of lidar data processing for autonomous driving and making the AI driver safer. More importantly, the proposed method is efficient to achieve the higher performance which can help decrease the carbon footprint and is thus environmental friendly.

As for the method itself, the fusion in the scale-pyramid should be explored more and we think we can remove part of the links to save more computational cost. Also, we have just validated the proposed network on several public datasets and haven't tested it on large datasets in real applications.

References

[1] Hasan Asy'ari Arief, Mansur Arief, Manoj Bhat, Ulf Geir Indahl, Håvard Tveite, and Ding Zhao. Density-adaptive sampling for heterogeneous point cloud object segmentation in autonomous vehicle applications. In *CVPR Workshops*, pages 26–33, 2019. 1

[2] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1534–1543, 2016. 5

[3] Alexandre Boulch. Convpoint: Continuous convolutions for point cloud processing. *Computers & Graphics*, 88:24–34, 2020. 8

[4] Alexandre Boulch, Gilles Puy, and Renaud Marlet. FKAConv: Feature-Kernel Alignment for Point Cloud Convolution. In *15th Asian Conference on Computer Vision (ACCV 2020)*, 2020. 8

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017. 1

[6] Lin-Zhuo Chen, Xuan-Yi Li, Deng-Ping Fan, Kai Wang, Shao-Ping Lu, and Ming-Ming Cheng. Lsanet: Feature learning on point sets by local spatial aware layer. *arXiv preprint arXiv:1905.05442*, 2019. 2

[7] Yimian Dai, Fabian Gieseke, Stefan Oehmcke, Yiquan Wu, and Kobus Barnard. Attentional feature fusion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3560–3569, 2021. 7

[8] Oren Dovrat, Itai Lang, and Shai Avidan. Learning to Sample. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2760–2769, 2019. 1

[9] Francis Engelmann, Theodora Kontogianni, Alexander Hermans, and Bastian Leibe. Exploring spatial context for 3d semantic segmentation of point clouds. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 716–724, 2017. 2

[10] Siqi Fan, Qiulei Dong, Fenghua Zhu, Yisheng Lv, Peijun Ye, and Fei-Yue Wang. Scf-net: Learning spatial contextual features for large-scale point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14504–14513, 2021. 8

[11] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847*, 2017. 5

[12] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 7

[13] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. 1, 2, 5, 6, 8

[14] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018. 1, 2

[15] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2626–2635, 2018. 2

[16] Q. Huang, W. Wang, and U. Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 8

[17] Mingyang Jiang, Yiran Wu, Tianqi Zhao, Zelin Zhao, and Cewu Lu. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv preprint arXiv:1807.00652*, 2018. 3

[18] Konstantinos Kamnitsas, Christian Ledig, Virginia FJ Newcombe, Joanna P Simpson, Andrew D Kane, David K Menon, Daniel Rueckert, and Ben Glocker. Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation. *Medical image analysis*, 36:61–78, 2017. 1

[19] Artem Komarichev, Zichun Zhong, and Jing Hua. A-cnn: Annularly convolutional neural networks on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7421–7430, 2019. 1

[20] Shiyi Lan, Ruichi Yu, Gang Yu, and Larry S Davis. Modeling local geometric structure of 3d point clouds using geo-cnn. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 998–1008, 2019. 1, 2

[21] Loic Landrieu and Mohamed Boussaha. Point cloud oversegmentation with graph-structured deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7440–7449, 2019. 2

[22] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4558–4567, 2018. 2

[23] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. 2018. 8

[24] Itai Lang, Asaf Manor, and Shai Avidan. Samplenet: differentiable point cloud sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7578–7588, 2020. 1

[25] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. 2018. 8

[26] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8759–8768, 2018. 2

[27] Ze Liu, Han Hu, Yue Cao, Zheng Zhang, and Xin Tong. A closer look at local aggregation operators in point cloud analysis. In *European Conference on Computer Vision*, pages 326–342. Springer, 2020. 1, 2, 4

[28] Tao Lu, Limin Wang, and Gangshan Wu. Cga-net: Category guided aggregation for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer*

Vision and Pattern Recognition, pages 11693–11702, 2021. 1

[29] Lingfei Ma, Ying Li, Jonathan Li, Weikai Tan, Yongtao Yu, and Michael A Chapman. Multi-scale point-wise convolutional neural networks for 3d object segmentation from lidar point clouds in large-scale environments. *IEEE Transactions on Intelligent Transportation Systems*, 2019. 3

[30] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1578–1587, 2019. 1, 2

[31] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928. IEEE, 2015. 1

[32] Dong Nie, Jia Xue, and Xiaofeng Ren. Bidirectional pyramid networks for semantic segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2020. 1, 2, 3

[33] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 1, 2, 8

[34] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 1, 2, 3

[35] Shi Qiu, Saeed Anwar, and Nick Barnes. Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1757–1767, 2021. 1

[36] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3577–3586, 2017. 1

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1, 2

[38] X. Roynard, J. E. Deschaud, and Franois Goulette. Classification of point cloud scenes with multiscale voxel deep network. 2018. 8

[39] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Paris-lille-3d: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. *The International Journal of Robotics Research*, 37(6):545–557, 2018. 5

[40] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2530–2539, 2018. 1

[41] Bo Sun and Philippos Mordohai. Oriented point sampling for plane detection in unorganized point clouds. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2917–2923. IEEE, 2019. 1

[42] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019. 3

[43] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. *arXiv preprint arXiv:1911.09070*, 2019. 3

[44] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*, 2020. 1

[45] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018. 1, 2

[46] H. Thomas, F Goulette, J. E. Deschaud, and B. Marcotegui. Semantic classification of 3d point clouds with multiscale spherical neighborhoods. In *2018 International Conference on 3D Vision (3DV)*, 2018. 8

[47] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6411–6420, 2019. 1, 2, 5, 6, 8

[48] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1, 2, 4

[49] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10296–10305, 2019. 2

[50] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 1

[51] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 7

[52] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 1

[53] Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3173–3182, 2021. 1

[54] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3323–3332, 2019. 1, 2

[55] Shuquan Ye, Dongdong Chen, Songfang Han, and Jing Liao. Learning with noisy labels for robust point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6443–6452, 2021. 1

[56] Xiaoqing Ye, Jiamao Li, Hexiao Huang, Liang Du, and Xiaolin Zhang. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 403–417, 2018. 2

[57] Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2403–2412, 2018. 1

[58] Zhiyuan Zhang, Binh-Son Hua, and Sai-Kit Yeung. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1607–1616, 2019. 1, 2

[59] H. Zhao et al. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017. 1

[60] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019. 1

[61] Kang Zhiheng and Li Ning. Pyramnet: Point cloud pyramid attention network and graph embedding module for classification and segmentation. *arXiv preprint arXiv:1906.03299*, 2019. 2