Semantic Segmentation for Real Point Cloud Scenes via Bilateral Augmentation and Adaptive Fusion

Shi Qiu[1,2], Saeed Anwar[1,2] and Nick Barnes[1]
[1]Australian National University [2]Data61-CSIRO
{shi.qiu, saeed.anwar, nick.barnes}@anu.edu.au

Abstract

Given the prominence of current 3D sensors, a fine-grained analysis on the basic point cloud data is worthy of further investigation. Particularly, real point cloud scenes can intuitively capture complex surroundings in the real world, but due to 3D data's raw nature, it is very challenging for machine perception. In this work, we concentrate on the essential visual task, semantic segmentation, for large-scale point cloud data collected in reality. On the one hand, to reduce the ambiguity in nearby points, we augment their local context by fully utilizing both geometric and semantic features in a bilateral structure. On the other hand, we comprehensively interpret the distinctness of the points from multiple resolutions and represent the feature map following an adaptive fusion method at point-level for accurate semantic segmentation. Further, we provide specific ablation studies and intuitive visualizations to validate our key modules. By comparing with state-of-the-art networks on three different benchmarks, we demonstrate the effectiveness of our network.

1. Introduction

As 3D data acquisition techniques develop rapidly, different types of 3D scanners, *e.g*. LiDAR scanners [22] and RGB-D cameras [10] are becoming popular in our daily life. Basically, 3D scanners can capture data that enables AI-driven machines to better see and recognize the world. As a fundamental data representation, point clouds can be easily collected using 3D scanners, retaining abundant information for further investigation. Therefore, point cloud analysis is playing an essential role in 3D computer vision.

Research has shown great success in terms of basic classification of small-scale point clouds (*i.e*., objects containing a few thousand points): for example, face ID [16] is now a widely used bio-identification for mobile devices. Researchers have recently been investigating a fine-grained analysis of large and complex point clouds [44, 26, 19, 48]

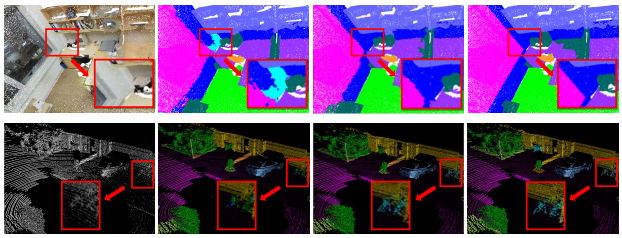


Figure 1: Examples of semantic segmentation for real point cloud scenes, where the main differences are highlighted and zoomed-in. The upper row shows an *indoor* working environment with ∼0.9 million points: RandLA-Net [19] falsely classifies the wall around the room corner, while our result is much closer to the ground-truth. The lower row is an *outdoor* traffic scene containing ∼32 thousand points, where a small bike on the right is correctly identified by our network (in blue), while RandLA-Net mislabels it as vegetation (in green).

because of the tremendous potential in applications such as autonomous driving, augmented reality, robotics, *etc*. This paper focuses on the semantic segmentation task to identify each point's semantic label for real point cloud scenes.

Although there are many notable works [41, 35, 55] addressing the semantic segmentation of 2D images which have a simpler structure, point clouds are *scattered, irregular, unordered, and unevenly distributed* in 3D space, making the corresponding task much more challenging, especially for large scenes made of millions or even billions of points collected from the real world. To deal with the 3D data, many papers try to build data-driven models using deep learning. Specifically, Guo *et al*. [13] summarizes the Convolutional Neural Network (CNN) models targeting point clouds into three streams: projection-based, discretization-based, and point-based methods. As a projection-based example, Lawin *et al*. [27] virtually projects 3D point clouds onto images and applies a conventional FCN [35] to analyze the 2D multi-view representations. Similarly, the discretization-based approaches model point clouds as voxels [20] or lattices [42] for CNN processing, and finally interpolate the semantic results back

to the original input. However, the mentioned methods are not optimal for real applications due to some common issues: firstly, they require several time-consuming pre/post-processing steps to make predictions; and secondly, the generated intermediate representations may partially lose the context of the surroundings.

To avoid the above issues, in this paper, we prefer point-based networks (details in Sec. 2) that directly process the points for fine-grained analysis. Moreover, for an accurate semantic segmentation on real point cloud scenes, we endeavor to resolve the major drawbacks of existing works:

Ambiguity in close points. Most current solutions [45, 11, 40] represent a point based on its pre-defined neighbors via a fixed metric like Euclidean distance. However, outliers and overlap between neighborhoods during the neighborhood's construction are difficult to avoid, especially when the points are closely distributed near the boundaries of different semantic classes. To alleviate possible impacts, we attempt to augment the local context by involving a dense region. Moreover, we introduce a robust aggregation process to refine the augmented local context and extract useful neighboring information for the point's representation.

Redundant features. We notice an increasing number of works [19, 50, 39] combine similar features multiple times to enhance the perception of the model. In fact, this process causes redundancy and increases the complexity for the model to process large-scale point clouds. To avoid the above problems, we propose to characterize the input information as geometric and semantic clues and then fully utilize them through a bilateral structure. More compactly, our design can explicitly represent complex point clouds.

Inadequate global representations. Although some approaches [38, 34, 29] apply an encoder-decoder [3] structure to learn the sampled point cloud; the output feature map is inadequate for a fine-grained semantic segmentation analysis since the global perception of the original data would be damaged during the sampling process. In our method, we intend to rebuild such perception by integrating information from different resolutions. Moreover, we adaptively fuse multi-resolutional features for each point to obtain a comprehensive representation, which can be directly applied for semantic prediction.

To conclude, our contributions are in these aspects:

- We introduce a bilateral block to augment the local context of the points.
- We adaptively fuse multi-resolutional features to acquire comprehensive knowledge about point clouds.
- We present a novel semantic segmentation network using our proposed structures to deal with real point cloud scenes.
- We evaluate our network on three large-scale benchmarks of real point cloud scenes. The experimental results demonstrate that our approach achieves competitive performances against state-of-the-art methods.

2. Related Work

Point-Based Approaches: As mentioned before, point-based approaches are designed to process unstructured 3D point cloud data directly rather than using its intermediate variants. Particularly, PointNet [37] applied the multi-layer-perceptron (MLP) and symmetric function (*e.g.*, max-pooling) to learn and aggregate point cloud features, respectively. Subsequently, point-wise MLPs were used to extract local features based on neighbor searching methods: *e.g.*, ball-query in PointNet++ [38], or k-nearest neighbors (knn) in DGCNN [45]. Moreover, MLPs were extended to perform point-convolutions: for instance, KPConv [44] leveraged kernel-points to convolve local point sets, while DPC [11] defined dilated point groups to increase the receptive fields of the points. Recurrent Neural Network (RNN) and Graph Convolutional Network (GCN) have also been adopted to replace regular CNNs in point-based approaches: for example, Liu *et al*. [33] transformed point clouds into sequences and processed the scaled areas using an LSTM structure, and Landrieu *et al*. [26] exploited super-point graphs to acquire semantic knowledge.

Point Clouds Feature Representations: Different from the individual point features in PointNet [37], the following methods focus on learning feature representations from local areas. Usually, the point neighbors are defined based on spatial metrics, *e.g.*, 3D Euclidean distances in [38, 34, 50, 19] or embedding similarities in [45, 39, 40]. By operating CNN-based modules over the neighborhoods, the local features of point clouds can be collected.

However, existing methods have limited capability to capture local details since they have not utilized the given information fully. Some works [37, 38, 45] only input the embedded features for each layer and lack the geometric restrictions in deep layers. Although current methods [34, 39] employ local descriptors to strengthen the spatial relations, however, the additional computational cost is involved. The latest approaches [50, 19] combine the original 3D coordinates in all scales of the network, but the effect is subtle. Differently, we exploit the point features from two properties: the geometric and semantic contexts. By augmenting them in a bilateral fashion, we can synthesize an augmented local context to represent the point.

Semantic Segmentation Networks: 2D semantic segmentation has been well studied in deep learning research. The basic FCN [35] applied a fully convolutional architecture to learn the features of each pixel. Further, UNet [41] designed the symmetric downsampling and upsampling structure for

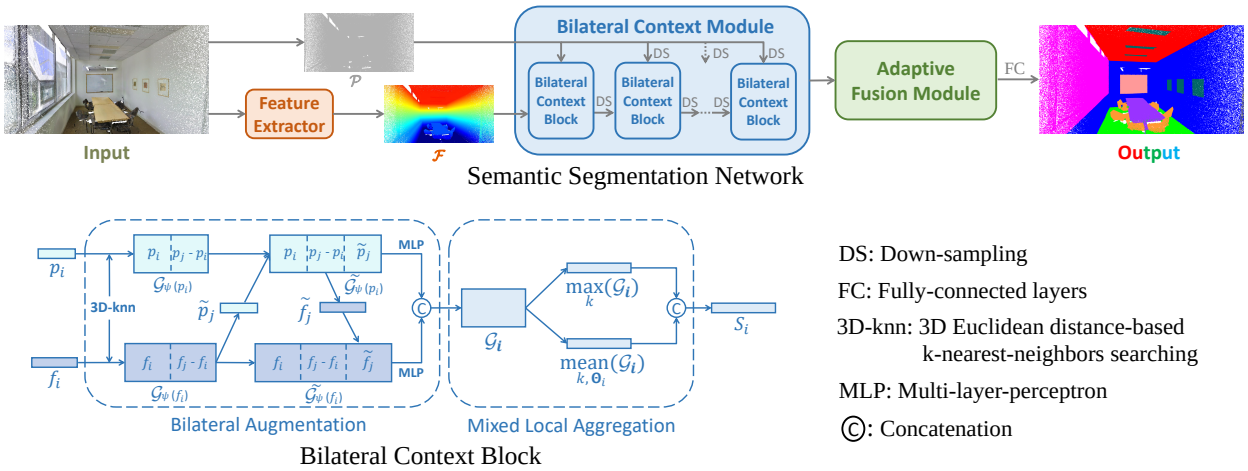


Figure 2: The details of our semantic segmentation network and the Bilateral Context Block (the annotations are consistent with the items in Sec. 3.1). Firstly, the Feature Extractor (Sec. 4.1) captures the preliminary semantic context $\mathcal{F}$ from the input data. Then, the Bilateral Context Module (*i.e.*, a series of the Bilateral Context Blocks) augments the local context of multiple point cloud resolutions. Generally, the Bilateral Context Block requires both semantic and geometric context as bilateral inputs. In particular, the first block inputs the preliminary $\mathcal{F}$ and the original 3D coordinates $\mathcal{P}$; while each of the rest inputs its previous one's downsampled output and coordinates $\mathcal{P}$, as the semantic and geometric context respectively. Afterward, our Adaptive Fusion Module (Sec. 3.2) upsamples the Bilateral Context Blocks' outputs, then adaptively fuses them as an output feature map. Finally, we predict semantic labels for all points via fully-connected layers.

image segmentation, while SegNet [3] proposed the convolutional encoder-decoder structure. More recently, Chen *et al*. [7] used a bi-directional gate to leverage multi-modality features, *i.e.*, color and depth, for RGB-D images.

In terms of 3D point clouds, most approaches are similar to the 2D image frameworks. For small-scale point clouds, the fully convolutional modules in [37, 45, 39] are able to manage the complexity of the data. In contrast, for large-scale data, some networks [38, 34, 19, 50] apply the convolutional encoder-decoders as SegNet [3] does, to generate the point-wise representations. However, the performance may be less satisfactory: as lower resolutions are explored, it becomes more difficult to interpret the local context of the unstructured 3D points. Although methods [11, 19, 40] attempt to tackle this problem by increasing the point's receptive field for a more detailed interpretation, it is expensive to find the optimal settings. Recent RandLA-Net [19] achieves high efficiency using naive random sampling, while the network's accuracy and stability are sacrificed. Unlike the existing methods, we propose a bilateral augmentation structure to effectively process multi-resolution point clouds, and utilize an adaptive fusion method to represent the comprehensive point-wise features efficiently.

3. Methodology

A point cloud containing N points can be described mainly from two aspects: 1) the inherent coordinates in 3D space $\mathcal{P} \in \mathbb{R}^{N\times 3}$ which are explicitly obtained by 3D scanners indicating the geometric context of the points; and 2) the acquired features $\mathcal{F} \in \mathbb{R}^{N\times d}$ in d-dimensional feature space which can be implicitly encoded by CNN-based operations implying latent clues about semantic context. From this point of view, $\mathcal{P}$ and $\mathcal{F}$ are regarded as two properties of the point cloud features.

Although $\mathcal{P}$ is less informative for semantic analysis, it can enrich the basic perception of geometry for the network. On this front, we aim to fully utilize $\mathcal{P}$ and $\mathcal{F}$ in a reasonable way, which can support learning a comprehensive feature map for accurate semantic segmentation.

3.1. Bilateral Context Module

The Bilateral Context Module consists of a number of Bilateral Context Blocks to investigate the point cloud at different resolutions, as shown in Fig. 2. In the Bilateral Context Block, we intend to augment the local context of each point by involving the offsets that are mutually learned from the bilateral input information (*i.e.*, $p_i \in \mathbb{R}^3$ and $f_i \in \mathbb{R}^d$), and then aggregate the augmented local context for the point feature representation. Particularly, we propose two novel units and a loss function to fulfill the intention.

Bilateral Augmentation: For a centroid p_i, we find its neighbors $\forall p_j \in Ni(p_i)$ using knn under the metric of 3D-Euclidean distance, while the corresponding neighbor features are denoted as f_j. To simultaneously capture both *global* and *local* information about the neighborhood, we combine the absolute position of the centroid and the relative positions of its neighbors as the local context $\mathcal{G}_\psi$. Accordingly, $\mathcal{G}_\psi(p_i) = [p_i; p_j - p_i]$ represents local geometric context in 3D space, while $\mathcal{G}_\psi(f_i) = [f_i; f_j - f_i]$ shows local semantic context in feature space.

However, $\mathcal{G}_\psi(p_i)$ and $\mathcal{G}_\psi(f_i)$ may be insufficient to represent the neighborhoods due to two reasons: 1) strict formation under a fixed constraint in 3D space could weaken the generalization capability of $\mathcal{G}_\psi$ in high-dimensional feature space, and 2) the $\mathcal{G}_\psi$ neighborhoods may have redundancy in the representations of close regions. To solve the issues and strengthen the generalization capability of the features, we can augment the local context by adding bilateral offsets, which shift the neighbors and densely affiliate them to the neighborhood's centroid.

To be specific, as the primary concern, we augment the local geometric context $\mathcal{G}_\psi(p_i)$ based on the rich semantic information of $\mathcal{G}_\psi(f_i)$. Particularly, we apply an MLP ($\mathcal{M}$) on $\mathcal{G}_\psi(f_i)$, to estimate the 3-DoF (Degrees of Freedom) bilateral offsets for the neighbors $\forall p_j \in Ni(p_i)$. Therefore, the shifted neighbors are formulated as:

$$\tilde{p_j} = \mathcal{M}(\mathcal{G}_\psi(f_i)) + p_j, \quad \tilde{p_j} \in \mathbb{R}^3. \tag{1}$$

Afterwards, we gather the *auxiliary* perception of the shifted neighbors to augment the local geometric context: $\tilde{\mathcal{G}_\psi}(p_i) = [p_i; p_j - p_i; \tilde{p_j}]$; where $\tilde{\mathcal{G}_\psi}(p_i) \in \mathbb{R}^{k\times 9}$ and k is the number of neighbors.

Subsequently, the d-DoF bilateral offsets for the neighbor features f_j can also be collected from $\tilde{\mathcal{G}_\psi}(p_i)$ since we expect the augmented local geometric context to further enhance the local semantic context. Similarly, the neighbor features are shifted as:

$$\tilde{f_j} = \mathcal{M}(\tilde{\mathcal{G}_\psi}(p_i)) + f_j, \quad \tilde{f_j} \in \mathbb{R}^d; \tag{2}$$

and the augmented local semantic context is formed as: $\tilde{\mathcal{G}_\psi}(f_i) = [f_i; f_j - f_i; \tilde{f_j}]$, where $\tilde{\mathcal{G}_\psi}(f_i) \in \mathbb{R}^{k\times 3d}$.

After further projecting the $\tilde{\mathcal{G}_\psi}(p_i)$ and $\tilde{\mathcal{G}_\psi}(f_i)$ by MLPs, we concatenate them as an augmented local context $\mathcal{G}_i$:

$$\mathcal{G}_i = \text{concat}\Big(\mathcal{M}\big(\tilde{\mathcal{G}_\psi}(p_i)\big), \mathcal{M}\big(\tilde{\mathcal{G}_\psi}(f_i)\big)\Big) \in \mathbb{R}^{k\times d'}. \tag{3}$$

Augmentation Loss: We also introduce some penalties to regulate the learning process of the bilateral offsets in Eq. 1. Since we should not only provide 3-DoF augmentation for the neighbors but also preserve the geometric integrity of a dense neighborhood, it is preferable to consider the neighbors as a whole instead of taking individual neighbors into account. Intuitively, we encourage the *geometric center* of the shifted neighbors to approach the local centroid in 3D space by minimizing the ℓ_2 distance:

$$\mathcal{L}(p_i) = \Big\| \tfrac{1}{k} \textstyle\sum_{j=1}^{k} \tilde{p_j} - p_i \Big\|_2. \tag{4}$$

Mixed Local Aggregation: Point-wise feature representation is crucial for the semantic segmentation task. Although non-parametric symmetric functions can efficiently summarize local information for the points, they cannot explicitly show the local distinctness, especially for close points sharing similar local context. To handle this problem, we propose a mixed local aggregation method to gather a precise neighborhood representation.

Algorithm 1: Adaptive Fusion Module Pipeline

input: M multi-resolution feature maps $\{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_M\}$.
output: $\mathcal{S}_{out}$ for semantic segmentation.
for $\mathcal{S}_m \in \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_M\}$ **do**
  upsample: $\tilde{\mathcal{S}_m} \leftarrow \mathcal{S}_m$;
  summarize: $\phi_m \leftarrow \tilde{\mathcal{S}_m}$;
end for
obtain: $\forall \tilde{\mathcal{S}_m} \in \{\tilde{\mathcal{S}_1}, \tilde{\mathcal{S}_2}, ..., \tilde{\mathcal{S}_M}\}$, $\tilde{\mathcal{S}_m} \in \mathbb{R}^{N\times c}$; and $\forall \phi_m \in \{\phi_1, \phi_2, ..., \phi_M\}$, $\phi_m \in \mathbb{R}^N$.
regress: $\{\Phi_1, \Phi_2, ..., \Phi_M\} \leftarrow \{\phi_1, \phi_2, ..., \phi_M\}$, where $\Phi_m \in \mathbb{R}^N$.
return: $\mathcal{S}_{out} = \sum_{m=1}^{M} \Phi_m \times \tilde{\mathcal{S}_m}$.

Given the augmented local context $\mathcal{G}_i$, on the one hand, we directly collect the *maximum* (prominent) feature from the k neighbors for an overview of the neighborhood. On the other hand, we closely investigate the representations of the neighbors, refining and obtaining more details by learning the high-dimensional barycenter (*i.e.*, weighted *mean* point) over the neighborhood. In the end, we combine the two types of information, the local *max* and *mean* features, to precisely represent the point as:

$$s_i = \text{concat}\Big(\max_{k}(\mathcal{G}_i), \underset{k,\Theta_i}{\text{mean}}(\mathcal{G}_i)\Big), \quad s_i \in \mathbb{R}^{2d'}; \tag{5}$$

where Θ_i is a set of learnable weights for k neighbors. The implementation details are in Sec. 4.2.

3.2. Adaptive Fusion Module

To efficiently analyze a real 3D scene consisting of a large number of points, we can gradually explore the point cloud in decreasing resolutions. Although it can be easily realized by applying the cascaded Bilateral Context Blocks for downsampled point cloud subsets, the corresponding output features become implicit and abstract. Therefore, it is essential to restore a feature map providing the original number of points and comprehensively interpret each point's encoded information. Specifically, we choose to fuse fine-grained representations from the multi-resolution feature maps adaptively.

Assume that M lower resolutions of the point cloud are processed by the Bilateral Context Module (*i.e.*, a cascaded set of the Bilateral Context Blocks as shown in Fig. 2), we extract a set of multi-resolution feature maps

as $\{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_M\}$ including $\{N_1, N_2, ..., N_M\}$ points, respectively.[1] As claimed in Alg. 1, for each extracted feature map $\forall \mathcal{S}_m \in \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_M\}$, we conduct progressive upsampling until a full-sized representation for all N points is generated. Following a similar process, we reconstruct the full-sized feature maps $\{\tilde{\mathcal{S}_1}, \tilde{\mathcal{S}_2}, ..., \tilde{\mathcal{S}_M}\}$.

Although we manage to interpret the whole point cloud, in terms of each point, the upsampled feature representations that originate from multiple resolutions may result in different scales of information. To integrate the information and refine the useful context for semantic segmentation, we fuse the full-sized feature maps adaptively at point-level.

To be concrete, we additionally summarize the point-level information $\phi_m \in \mathbb{R}^N$ during the upsampling process of each full-sized feature map's generation, in order to capture basic point-level understanding from different scales. Next, by analyzing those point-level perceptions $\{\phi_1, \phi_2, ..., \phi_M\}$ on the whole, we regress the fusion parameters $\{\Phi_1, \Phi_2, .., \Phi_M\}$ corresponding to the full-sized feature maps $\{\tilde{\mathcal{S}_1}, \tilde{\mathcal{S}_2}, ..., \tilde{\mathcal{S}_M}\}$, respectively. In the end, a comprehensive feature map $\mathcal{S}_{out}$ for semantic segmentation is adaptively fused throughout multi-resolution features *w.r.t.* each point. Theoretically, it follows:

$$\mathcal{S}_{out} = \sum_{m=1}^{M} \Phi_m \times \tilde{\mathcal{S}_m}, \quad \Phi_m \in \mathbb{R}^N. \tag{6}$$

More details about the Adaptive Fusion Module implementation are presented in Sec. 4.3.

4. Implementation Details

Based on the key structures in Sec. 3, we form an effective network for the semantic segmentation of real point clouds scenes. As illustrated in Fig. 2, our network has three modules: the Feature Extractor, the Bilateral Context Module, and the Adaptive Fusion Module. We introduce the details of each module in the following sections.

4.1. Feature Extractor

Besides the spatial 3D coordinates, some datasets may include other clues, *e.g.*, RGB colors, light intensity, *etc*. To create an overall impression of the whole scene, initially, we apply the Feature Extractor to acquire basic semantic knowledge from all of the provided information. Given the advantages of MLP that it can represent the features flexibly in high-dimensional embedding space, empirically, we apply a single-layer MLP (*i.e.*, a 1-by-1 convolutional layer followed by a batch normalization [21] and an activation function like ReLU) to obtain high-level compact features.

[1] $N > N_1 > N_2 > ... > N_M$, N is the original size of a point cloud.

Fig. 2 shows the acquired features $\mathcal{F}$ from the Feature Extractor which are forwarded to the Bilateral Context Module, along with the 3D coordinates $\mathcal{P}$.

4.2. Bilateral Context Module Implementation

As mentioned before, the Bilateral Context Module explores the different resolutions of point cloud data. For the sake of stability, we use CUDA-based Farthest Point Sampling (FPS) to sample the data based on its 3D distribution. Particularly, the Bilateral Context Module deploys cascaded Bilateral Context Blocks to gradually process the lower resolutions of the point cloud: *e.g.*, $N \rightarrow \frac{N}{4} \rightarrow \frac{N}{16} \rightarrow \frac{N}{64} \rightarrow \frac{N}{256}$. Meanwhile, the dimensions of the outputs are increasing as: $32 \rightarrow 128 \rightarrow 256 \rightarrow 512 \rightarrow 1024$. In this regard, the behavior of the Bilateral Context Module processing the 3D point clouds is similar to the classical CNNs for 2D images, which extend the channel number while shrinking the image size for a concise description.

Inside each Bilateral Context Block, an efficient k-nearest neighbor using the nanoflann [5] library speeds up neighbor searching in the bilateral augmentation unit. Empirically, we set k=12 for all experiments in this work. For the mixed local aggregation unit, the local *max* feature is collected by operating a max-pooling function along the neighbors. Following a similar operation in [19], we simultaneously refine and re-weight the neighbors through a single-layer MLP and a $\mathrm{softmax}$ function, then aggregate the barycenter of local embeddings as the local *mean* feature. Finally, the local *max* and *mean* features are concatenated as the output of the mixed local aggregation unit.

4.3. Adaptive Fusion Module Implementation

As explained in Sec. 3.2, our Adaptive Fusion Module aims to upsample the multi-resolution outputs of the Bilateral Context Module, and then adaptively fuse them as a comprehensive feature map for the whole point cloud scene. To be more specific with the upsampling process, at first, a single-layer MLP integrates the channel-wise information of the output feature maps. Then, we point-wisely interpolate a higher-resolution feature map using nearest neighbor interpolation [23], since it is more efficient for large-scale data than Feature Propagation [38] that requires huge computational cost for neighbors and weights. Moreover, we symmetrically attach the features from the same resolution in order to increase diversity and distinctness for nearby points. Finally, a higher-resolution feature map is synthesized via another single-layer MLP.

The upsampling process is continuously performed to get full-sized feature maps $\{\tilde{\mathcal{S}_1}, \tilde{\mathcal{S}_2}, ..., \tilde{\mathcal{S}_M}\}$ from the multi-resolution outputs of the Bilateral Context Module. During this process, we also use a fully-connected layer to summarize the point-level information ϕ_m once a full-sized feature map $\tilde{\mathcal{S}_m}$ is reconstructed. To analyze the summarized

information, we concatenate $\{\phi_1, \phi_2, ..., \phi_M\}$, and point-wisely normalize them using $\mathrm{softmax}$. As a result, the fusion parameters $\{\Phi_1, \Phi_2, ..., \Phi_M\}$ are adaptively regressed *w.r.t.* each point. After calculating a weighted sum of the upsampled feature maps (Eq. 6), we eventually combine a feature map containing all points for whole scene semantic segmentation. Besides, a structure chart of this module is provided in the supplementary material.

4.4. Loss Function

Using the fused output of the Adaptive Fusion Module, the FC layers predict the confidence scores for all candidate semantic classes. Generally, cross-entropy loss $\mathcal{L}_{CE}$ is computed for back-propagation. Further, we include point-level augmentation losses $\mathcal{L}(p_i)$ that are formulated following Eq. 4. In terms of a Bilateral Context Block processing N_m points, the total augmentation loss regarding N_m points would be $\mathcal{L}_m = \sum_{i=1}^{N_m} \mathcal{L}(p_i)$. Hence, for our network containing M Bilateral Context Blocks, the overall loss is:

$$\mathcal{L}_{all} = \mathcal{L}_{CE} + \sum_{m=1}^{M} \omega_m \cdot \mathcal{L}_m, \tag{7}$$

where ω_m is a hyper-parameter of weight for each Bilateral Context Block.

5. Experiments

5.1. Experimental Settings

Datasets: In this work, we are targeting the semantic segmentation of real point cloud scenes. To validate our approach, we conduct experiments on three 3D benchmarks, which present different scenes in the real world.

- **S3DIS:** Stanford Large-Scale 3D Indoor Spaces (S3DIS) [2] dataset is collected from *indoor* working environments. In general, there are six sub-areas in the dataset, each containing ∼50 different rooms. The number of points in most rooms varies from 0.5 million to 2.5 million, depending on the room's size. All points are provided with both 3D coordinates and color information and labeled as one of 13 semantic categories. We adopt a 6-fold strategy [37] for evaluation.
- **Semantic3D:** The points in Semantic3D [14] are scanned in *natural* scenes depicting various rural and urban views. Overall, this dataset contains more than four billion points manually marked in eight semantic classes. In particular, the dataset has two test sets for online evaluation: the full test set (*i.e.*, semantic-8) has 15 scenes with over 2 billion points, while its subset (*i.e.*, reduced-8) has four selected scenes with ∼0.1 billion sampled points. In this work, we use both 3D positions and colors of points for training and then infer the dense scenes of entire *semantic-8* test set.

Table 1: Semantic segmentation (6-fold cross-validation) results (%) on the *S3DIS* dataset [2]. (**mAcc**: average class accuracy, **OA**: overall accuracy, **mIoU**: mean Intersection-over-Union. "-" indicates unknown result.)

year	Method	**mAcc**	**OA**	**mIoU**
2017	PointNet [37]	66.2	78.6	47.6
	PointNet++ [38]	67.1	81.0	54.5
2018	A-SCN [49]	-	81.6	52.7
	PointCNN [30]	75.6	88.1	65.4
	SPG [26]	73.0	85.5	62.1
2019	DGCNN [45]	-	84.1	56.1
	KP-Conv [44]	79.1	-	70.6
	ShellNet [53]	-	87.1	66.8
	PointWeb [54]	76.2	87.3	66.7
	SSP+SPG [25]	78.3	87.9	68.4
2020	Seg-GCN [28]	77.1	87.8	68.5
	PointASNL [50]	79.0	88.8	68.7
	RandLA-Net [19]	82.0	88.0	70.0
	MPNet [17]	-	86.8	61.3
	InsSem-SP [32]	74.3	88.5	64.1
	Ours	**83.1**	**88.9**	**72.2**

- **SemanticKITTI:** SemanticKITTI [4] was introduced based on the well-known KITTI Vision [12] benchmark illustrating complex *outdoor* traffic scenarios. There are 22 stereo sequences, which are densely recorded as scans (∼0.1 million points in each scan) and precisely annotated in 19 semantic classes. Particularly, 11 of the 22 sequences are provided with labels, while the results of the other ten sequences (over 20k scans) are for online evaluation. As in [4], we take sequence 08 as the validation set, while the remaining ten labeled sequences (∼19k scans) are for training.

Training Settings: We train for 100 epochs on a single GeForce RTX 2080Ti GPU with a batch size between 4 to 6, depending on the amount of input points (about 40×2^{10} to 64×2^{10}) for different datasets. In addition, the Adam [24] optimizer is employed to minimize the overall loss in Eq. 7; the learning rate starts from 0.01 and decays with a rate of 0.5 after every 10 epochs. We implement the project[2] in Python and Tensorflow [1] platforms using Linux.

Evaluation Metrics: To evaluate our semantic segmentation performance, we largely use the mean Intersection-over-Union (mIoU), the average value of IoUs for all semantic classes upon the whole dataset. Further, we also provide the overall accuracy (OA) regarding all points and the average class accuracy (mAcc) of all semantic classes. As for S3DIS [2], we compute the mIoU based on all predicted sub-areas following the 6-fold strategy. Similarly, for both Semantic3D [14] and SemanticKITTI [4], we provide the online submission testing results of general mIoU and

[2]The codes and test results are available at https://github.com/ShiQiu0419/BAAF-Net.

Table 2: Semantic segmentation (semantic-8) results (%) on the *Semantic3D* dataset [14].

Method	**OA**	**mIoU**	man-made terrain	natural terrain	high vegetation	low vegetation	buildings	hard scape	scanning artefacts	cars
TMLC-MS [15]	85.0	49.4	91.1	69.5	32.8	21.6	87.6	25.9	11.3	55.3
EdgeConv-PN [9]	89.4	61.0	91.2	69.8	51.4	58.5	90.6	33.0	24.9	68.6
PointNet++ [38]	85.7	63.1	81.9	78.1	64.3	51.7	75.9	36.4	43.7	72.6
SnapNet [6]	91.0	67.4	89.6	79.5	74.8	56.1	90.9	36.5	34.3	77.2
PointConv [47]	91.8	69.2	92.2	79.2	73.1	62.7	92.0	28.7	43.1	82.3
PointGCR [36]	92.1	69.5	93.8	80.0	64.4	66.4	93.2	39.2	34.3	85.3
PointConv-CE [31]	92.3	71.0	92.4	79.6	72.7	62.0	93.7	40.6	44.6	82.5
RandLA-Net [19]	94.2	71.8	96.0	88.6	65.3	62.0	**95.9**	49.8	27.8	89.3
SPG [26]	92.9	**76.2**	91.5	75.6	**78.3**	**71.7**	94.4	**56.8**	**52.9**	88.4
Ours	**94.9**	75.4	**97.9**	**95.0**	70.6	63.1	94.2	41.6	50.2	**90.3**

Table 3: Semantic segmentation (single-scan) results (%) on the *SemanticKITTI* dataset [4].

Method	**mIoU**	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic-sign
PointNet [37]	14.6	61.6	35.7	15.8	1.4	41.4	46.3	0.1	1.3	0.3	0.8	31.0	4.6	17.6	0.2	0.2	0.0	12.9	2.4	3.7
PointNet++ [38]	20.1	72.0	41.8	18.7	5.6	62.3	53.7	0.9	1.9	0.2	0.2	46.5	13.8	30.0	0.9	1.0	0.0	16.9	6.0	8.9
SquSegV2 [46]	39.7	88.6	67.6	45.8	17.7	73.7	81.8	13.4	18.5	17.9	14.0	71.8	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.3
TangentConv [43]	40.9	83.9	63.9	33.4	15.4	83.4	90.8	15.2	2.7	16.5	12.1	79.5	49.3	58.1	23.0	28.4	8.1	49.0	35.8	28.5
PointASNL [50]	46.8	87.4	74.3	24.3	1.8	83.1	87.9	39.0	0.0	25.1	29.2	84.1	52.2	**70.6**	34.2	**57.6**	0.0	43.9	57.8	36.9
RandLA-Net [19]	53.9	90.7	73.7	60.3	20.4	86.9	94.2	40.1	26.0	25.8	38.9	81.4	61.3	66.8	49.2	48.2	7.2	56.3	49.2	47.7
PolarNet [52]	54.3	90.8	74.4	61.7	21.7	90.0	93.8	22.9	40.3	30.1	28.5	84.0	65.5	67.8	43.2	40.2	5.6	67.8	51.8	57.5
MinkNet42 [8]	54.3	91.1	69.7	63.8	29.3	**92.7**	94.3	26.1	23.1	26.2	36.7	83.7	68.4	64.7	43.1	36.4	7.9	57.1	57.3	60.1
FusionNet [51]	**61.3**	**91.8**	**77.1**	**68.8**	**30.8**	92.5	95.3	41.8	**47.5**	**37.7**	34.5	**84.5**	**69.8**	68.5	**59.5**	56.8	11.9	**69.4**	**60.4**	**66.5**
Ours	59.9	90.9	74.4	62.2	23.6	89.8	**95.4**	**48.7**	31.8	35.5	**46.7**	82.7	63.4	67.9	49.5	55.7	**53.0**	60.8	53.7	52.0

Table 4: Ablation studies about the Bilateral Context Block testing on Area 5, *S3DIS* dataset. ($\tilde{p}_i \rightarrow \tilde{f}_i$: learn 3-DoF offsets $\tilde{p}_i$ first and d-DoF offsets $\tilde{f}_i$ afterwards as per Sec. 3.1; $\tilde{f}_i \rightarrow \tilde{p}_i$: learn $\tilde{f}_i$, and then $\tilde{p}_i$; $\mathcal{L}(\cdot)$: calculate augmentation loss as per Eq. 4; *mixed*: mixed local aggregation following Eq. 5; *max*: local max feature $\max_k(\mathcal{G}_i)$ only; *mean*: local mean feature $\underset{k,\Theta_i}{\text{mean}}(\mathcal{G}_i)$ only.)

Model	bilateral offsets	augmentation loss	local aggregation	mIoU
B_0	none	none	max	61.8
B_1	$\tilde{f}_i \rightarrow \tilde{p}_i$	$\mathcal{L}(f_i)$	mixed	64.2
B_2	$\tilde{p}_i \rightarrow \tilde{f}_i$	$\mathcal{L}(p_i) + \mathcal{L}(f_i)$	mixed	64.3
B_3	$\tilde{p}_i \rightarrow \tilde{f}_i$	none	mixed	64.2
B_4	$\tilde{p}_i \rightarrow \tilde{f}_i$	$\mathcal{L}(p_i)$	max	64.6
B_5	$\tilde{p}_i \rightarrow \tilde{f}_i$	$\mathcal{L}(p_i)$	mean	64.8
$\mathbf{B_6}$	$\tilde{p}_i \rightarrow \tilde{f}_i$	$\mathcal{L}(p_i)$	mixed	**65.4**

Table 5: Ablation studies about the Adaptive Fusion Module testing on Area 5, *S3DIS* dataset. ($\{\tilde{\mathcal{S}_m}\}$: a set of upsampled feature maps, $\tilde{\mathcal{S}_1}$,..,$\tilde{\mathcal{S}_M}$, as mentioned in Alg. 1; concat, $\sum$ and $\prod$: the concatenation, element-wise sum and element-wise multiplication for the set $\{\tilde{\mathcal{S}_m}\}$; $\{\Psi_m\}$: scalars for the set $\{\tilde{\mathcal{S}_m}\}$; $\{\Phi_m\}$: point-level fusion parameters as explained in Sec. 3.2 and 4.3.)

Model	upsampled feature map	fusion parameters	$\mathcal{S}_{out}$	mIoU
A_0	$\tilde{\mathcal{S}_M}$	none	$\tilde{\mathcal{S}_M}$	64.1
A_1	$\{\tilde{\mathcal{S}_m}\}$	none	$\sum \tilde{\mathcal{S}_m}$	64.7
A_2	$\{\tilde{\mathcal{S}_m}\}$	none	$\prod \tilde{\mathcal{S}_m}$	64.2
A_3	$\{\tilde{\mathcal{S}_m}\}$	none	$\text{concat}(\{\tilde{\mathcal{S}_m}\})$	65.1
A_4	$\{\tilde{\mathcal{S}_m}\}$	$\{\Psi_m\}$	$\sum \Psi_m \times \tilde{\mathcal{S}_m}$	65.1
$\mathbf{A_5}$	$\{\tilde{\mathcal{S}_m}\}$	$\{\Phi_m\}$	$\sum \Phi_m \times \tilde{\mathcal{S}_m}$	**65.4**

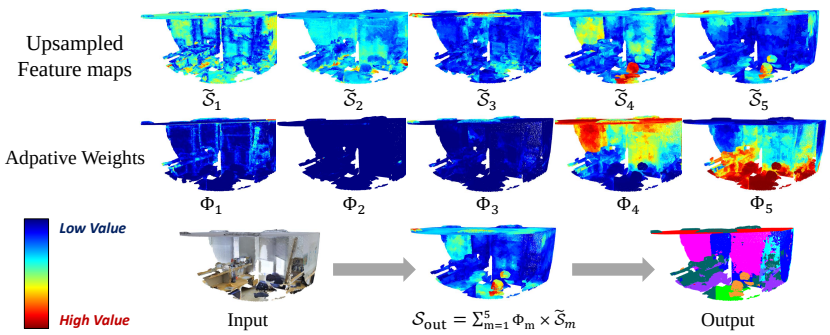
OA, as well as the IoU for each semantic category.

5.2. Semantic Segmentation Results

S3DIS: Tab. 1 quantitatively presents the performance of our network on the S3DIS dataset compared with other state-of-the-art methods. Notably, although recent methods achieve good results regarding overall accuracy, this metric is unable to indicate the semantic segmentation ability due to class imbalance among different categories. In general, we significantly outperform the competitors regarding the metrics of average class accuracy (83.1%) and mIoU (72.2%). Moreover, we visualize the Adaptive Fusion Module's upsampled features maps and adaptive weights in Fig. 3 (better in a zoom-in and colored view) based on S3DIS, in order to intuitively analyze the module's behavior while fusing the multi-resolution feature maps.

Semantic3D: We also perform well on the natural views of the Semantic3D dataset. As Tab. 2 indicates, we surpass other methods in three out of the eight classes; and our method is accurate on three categories, *i.e.*, *human-made and natural terrains, cars*, whose IoUs are all higher than 90%. Considering the results of both overall accuracy (94.9%) and mIoU (75.4%) upon two billion testing points, our method accurately classifies the semantic labels of points in real scenes, especially for large-scale data.

SemanticKITTI: Although SemanticKITTI is challenging due to the complex scenarios in traffic environments,



The 1st row shows the feature maps that are upsampled from multiple resolutions, where the larger subscript number denotes the one from a lower resolution (*i.e.*, a deeper layer). We average and normalize the feature map's channels to illustrate them in a form of heat-map.

The 2nd row presents the element-wise fusion weights. The feature maps from higher resolutions (shallow layers) focus on some simple features: $\widetilde{\mathcal{S}}_1$ is about the edges/frames, and Φ_1 strengthens them with higher weights; $\widetilde{\mathcal{S}}_2$ cares more about the corners, but Φ_2 assigns low weights since they are semantically trivial; $\widetilde{\mathcal{S}}_3$ and Φ_3 concentrate on small objects on the ceiling. In contrast, the feature maps upsampled from lower resolutions (deep layers) can gather more semantically meaningful information from different scales: $\widetilde{\mathcal{S}}_4$ puts much attention Φ_4 on the upper half of the office, as it differentiates the *beam*, *wall* and *ceiling*; while Φ_5 contributes more to the lower half, as $\widetilde{\mathcal{S}}_5$ clearly separates *chair*, *table*, *clutter* and *floor*.

Figure 3: Behavior analysis of the Adaptive Fusion Module (Sec. 3.2) based on an office scene in *S3DIS* dataset. By fusing the upsampled feature maps in a simple but adaptive way, we aggregate the advantages from different scales, and generate $\mathcal{S}_{out}$ for semantic segmentation.

Table 6: Complexity analysis of different semantic segmentation networks on *SemanticKITTI*. ("-" indicates the unknown result.)

Method	Parameters (*millions*)	Max Capacity (*million points*)	Inference Speed (*scans/second*)	mIoU
PointNet [37]	0.8	0.49	21.2	14.6
PointNet++ [38]	0.97	0.98	0.4	20.1
SPG [26]	**0.25**	-	0.1	17.4
RandLA-Net [19]	1.24	**1.03**	**22**	53.9
Ours	1.23	0.9	4.8	**59.9**

our network can effectively identify the semantic labels of points. As shown in Tab. 3, we exceed other advanced approaches in 4 of all 19 classes. Particularly, we perform well regarding the *small* objects in dense scans such as *car, truck, other-vehicle, motorcyclist*, *etc*. The outstanding results can be credited to our point-level adaptive fusion method, which thoroughly integrates the different scales. Overall, our network boosts a lot (5.6% mIoU) compared to the latest point and grid-based methods [50, 19, 52], while is slightly behind the state-of-the-art work [51] using sparse tensor-based framework [8]. As our main ideas of bilateral augmentation and adaptive fusion are fairly adaptable, more experiments with different frameworks will be studied in the future.

5.3. Ablation Studies

Bilateral Context Block: In Tab. 4, we study the Bilateral Context Block's structure by investigating the components individually. B_0 is the baseline model which only max-pools the concatenation of the basic local geometric $\mathcal{G}_\psi(p_i)$ and semantic context $\mathcal{G}_\psi(f_i)$; while rest models use different components based on the same structure of bilateral augmentation. From model B_1&B_2, we observe that the semantic augmentation loss $\mathcal{L}(f_i)$ has no effect since augmenting the semantic features in embedding space is implicit. In contrast, the bilateral offsets $\tilde{p}_i$ with the geometric augmentation loss $\mathcal{L}(p_i)$ improves a bit (model B_4&B_5). Taking the advantages from both local *max* and *mean* features, we conclude that the best form of the Bilateral Context Block is using *mixed* local aggregation (B_6).

Adaptive Fusion Module: In Tab. 5, by comparing models A_1, A_2&A_3 with the baseline A_0 that only upsamples the final output of the Bilateral Context Module, we notice that utilizing the upsampled features maps that originate from multiple resolutions can benefit the performance. However, the fusion method decides whether the effects are significant or not: regular summation (A_1) or multiplication (A_2) is not desirable, while concatenation (A_3) contributes more to the final prediction. For a general fusion (A_4) *w.r.t.* each feature map, we regress a set of scalars $\{\Psi_m\}$ based on the squeezed information [18] of the feature maps. Instead, a more flexible fusion operating adaptively at point-level (A_5) achieves better results since semantic segmentation relies more on point-wise feature representations.

Network Complexity: Network complexity is essential to the practical application of point clouds. In Tab. 6, we use similar metrics as [19] to study the inference using the trained models. The complexity and capacity (*i.e.*, the number of parameters, and the maximum number of points for prediction) of our model are comparable to [38, 19]. Although [19] is efficient for one-time inference, they require multiple evaluations to minimize the impact of random sampling, while we obtain more effective and stable semantic segmentation results in different real scenes such as the examples shown in Fig. 1. More visualizations and experimental results are presented in the supplementary material.

6. Conclusions

This paper focuses on fundamental analysis and semantic segmentation for real point clouds scenes. Specifically, we propose a network leveraging the ideas of augmenting the local context bilaterally and fusing multi-resolution features for each point adaptively. Particularly, we achieve outstanding performance on three benchmarks, including S3DIS, Semantic3D, and SemanticKITTI. Further, we analyze the modules' properties by conducting related ablation studies, and intuitively visualize the network's effects. In the future, we expect to optimize the efficiency for real-time applications, exploit the key ideas in different frameworks, and promote the primary structures for more 3D tasks such as object detection, instance segmentation, *etc*.

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283, 2016. 6

[2] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 6, 13, 14, 16

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017. 2, 3

[4] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9297–9307, 2019. 6, 7, 18

[5] Jose Luis Blanco and Pranjal Kumar Rai. nanoflann: a C++ header-only fork of FLANN, a library for nearest neighbor (NN) with kd-trees. https://github.com/jlblancoc/nanoflann, 2014. 5

[6] Alexandre Boulch, Joris Guerry, Bertrand Le Saux, and Nicolas Audebert. Snapnet: 3d point cloud semantic labeling with 2d deep segmentation networks. *Computers & Graphics*, 71:189–198, 2018. 7, 13

[7] Xiaokang Chen, Kwan-Yee Lin, Jingbo Wang, Wayne Wu, Chen Qian, Hongsheng Li, and Gang Zeng. Bi-directional cross-modality feature propagation with separation-and-aggregation gate for rgb-d semantic segmentation. In *European Conference on Computer Vision*, pages 644–663. Springer, 2020. 3

[8] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3075–3084, 2019. 7, 8

[9] Jhonatan Contreras and Joachim Denzler. Edge-convolution point net for semantic segmentation of large-scale point clouds. In *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*, pages 5236–5239. IEEE, 2019. 7

[10] Felix Endres, Jürgen Hess, Jürgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-d mapping with an rgb-d camera. *IEEE transactions on robotics*, 30(1):177–187, 2013. 1

[11] Francis Engelmann, Theodora Kontogianni, and Bastian Leibe. Dilated point convolutions: On the receptive field size of point convolutions on 3d point clouds. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9463–9469. IEEE, 2020. 2, 3, 13, 14

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 6

[13] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2020. 1

[14] 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. 6, 7, 13, 17

[15] Timo Hackel, Jan D Wegner, and Konrad Schindler. Fast semantic segmentation of 3d point clouds with strongly varying density. *ISPRS annals of the photogrammetry, remote sensing and spatial information sciences*, 3:177–184, 2016. 7

[16] Rida M Hamza, Michael E Bazakos, and Murray J Cooper. Face identification verification using 3 dimensional modeling, Sept. 2 2008. US Patent 7,421,097. 1

[17] Tong He, Dong Gong, Zhi Tian, and Chunhua Shen. Learning and memorizing representative prototypes for 3d point cloud semantic and instance segmentation. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 6

[18] 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. 8

[19] 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, 3, 5, 6, 7, 8, 12, 13

[20] Jing Huang and Suya You. Point cloud labeling using 3d convolutional neural network. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2670–2675. IEEE, 2016. 1

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 5

[22] Michel Jaboyedoff, Thierry Oppikofer, Antonio Abellán, Marc-Henri Derron, Alex Loye, Richard Metzger, and Andrea Pedrazzini. Use of lidar in landslide investigations: a review. *Natural hazards*, 61(1):5–28, 2012. 1

[23] Robert Keys. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, 29(6):1153–1160, 1981. 5

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[25] Loic Landrieu and Mohamed Boussaha. Point cloud oversegmentation with graph-structured deep metric learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7440–7449, 2019. 6

[26] 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. 1, 2, 6, 7, 8, 13

[27] Felix Järemo Lawin, Martin Danelljan, Patrik Tosteberg, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg.

Deep projective 3d semantic segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 95–107. Springer, 2017. 1

[28] Huan Lei, Naveed Akhtar, and Ajmal Mian. Seggcn: Efficient 3d point cloud segmentation with fuzzy spherical kernel. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11611–11620, 2020. 6

[29] Ruihui Li, Xianzhi Li, Chi-Wing Fu, Daniel Cohen-Or, and Pheng-Ann Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7203–7212, 2019. 2

[30] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018. 6

[31] H. Liu, Y. Guo, Y. Ma, Y. Lei, and G. Wen. Semantic context encoding for accurate 3d point cloud segmentation. *IEEE Transactions on Multimedia*, pages 1–1, 2020. 7

[32] Jinxian Liu, Minghui Yu, Bingbing Ni, and Ye Chen. Self-prediction for joint instance and semantic segmentation of point clouds. In *European Conference on Computer Vision*, pages 187–204. Springer, 2020. 6

[33] Xinhai Liu, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8778–8785, 2019. 2

[34] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019. 2, 3

[35] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1, 2

[36] Yanni Ma, Yulan Guo, Hao Liu, Yinjie Lei, and Gongjian Wen. Global context reasoning for semantic segmentation of 3d point clouds. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 2931–2940, 2020. 7

[37] 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. 2, 3, 6, 7, 8

[38] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 2, 3, 5, 6, 7, 8

[39] Shi Qiu, Saeed Anwar, and Nick Barnes. Geometric back-projection network for point cloud classification. *arXiv preprint arXiv:1911.12885*, 2019. 2, 3

[40] Shi Qiu, Saeed Anwar, and Nick Barnes. Dense-resolution network for point cloud classification and segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3813–3822, January 2021. 2, 3

[41] 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

[42] 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

[43] 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. 7

[44] 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 International Conference on Computer Vision*, pages 6411–6420, 2019. 1, 2, 6, 13

[45] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):146, 2019. 2, 3, 6

[46] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE, 2019. 7

[47] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019. 7

[48] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 1

[49] Saining Xie, Sainan Liu, Zeyu Chen, and Zhuowen Tu. Attentional shapecontextnet for point cloud recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4606–4615, 2018. 6

[50] Xu Yan, Chaoda Zheng, Zhen Li, Sheng Wang, and Shuguang Cui. Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5589–5598, 2020. 2, 3, 6, 7, 8

[51] Feihu Zhang, Jin Fang, Benjamin Wah, and Philip Torr. Deep fusionnet for point cloud semantic segmentation. In *European Conference on Computer Vision*, pages 644–663. Springer, 2020. 7, 8

[52] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. Polarnet: An

improved grid representation for online lidar point clouds semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9601–9610, 2020. 7, 8

[53] 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 International Conference on Computer Vision*, pages 1607–1616, 2019. 6, 13

[54] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019. 6

[55] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017. 1

Supplementary Material

A. Overview

This supplementary material provides more network details, experimental results, and visualizations of our semantic segmentation results.

B. Network Details

In Fig. 2 of the main paper, we present the general architecture of our semantic segmentation network as well as the structure of the Bilateral Context Block. In this section, we provide more details about the different components of our network.

B.1. Key Modules

Feature Extractor: As stated, we apply a single-layer MLP containing eight 1×1 kernels to extract the semantic context $\mathcal{F}$ from the input information $\mathcal{I} \in \mathbb{R}^{N\times C_{in}}$, where N is the number of input points. Hence, $\mathcal{F}$ is acquired as:

$$\mathcal{F} = \mathrm{ReLU}\Big(\mathrm{BN}\big(\mathrm{Conv}^{8}_{1\times1}(\mathcal{I})\big)\Big), \quad \mathcal{F} \in \mathbb{R}^{N\times 8};$$

where Conv denotes a convolution layer whose subscript is the kernel size, and the superscript is the number of kernels. BN represents a batch normalization layer, while ReLU is a ReLU activation layer. Later on, $\mathcal{F}$ is forwarded to the Bilateral Context Module, together with the 3D coordinates $\mathcal{P} \in \mathbb{R}^{N\times 3}$.

Bilateral Context Module: In practice, we apply five Bilateral Context Blocks with Farthest Point Sampling (FPS) to realize the Bilateral Context Module ($\mathcal{B}$). Using the same annotations of the main paper's Sec. 4.2, the extracted multi-resolution feature maps are:

$$\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4, \mathcal{S}_5\} = \mathcal{B}(\mathcal{P}, \mathcal{F});$$

where:

$$\mathcal{S}_1 \in \mathbb{R}^{\frac{N}{4}\times 32}, \quad \mathcal{S}_2 \in \mathbb{R}^{\frac{N}{16}\times 128}, \quad \mathcal{S}_3 \in \mathbb{R}^{\frac{N}{64}\times 256},$$

$$\mathcal{S}_4 \in \mathbb{R}^{\frac{N}{256}\times 512}, \quad \mathcal{S}_5 \in \mathbb{R}^{\frac{N}{512}\times 1024}.$$

Particularly, the downsampling ratios and feature dimensions are simply adopted from [19], since we mainly focus on the structure design rather than fine-tuning the hyper-parameters in this work.

Adaptive Fusion Module: In addition to Alg. 1 and Sec. 3.2 in the main paper, we also illustrate the architecture of the Adaptive Fusion Module in Fig. 4 as a complement. As described in Sec. 4.3 of the main paper, we gradually upsample the extracted feature maps $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4, \mathcal{S}_5\}$,

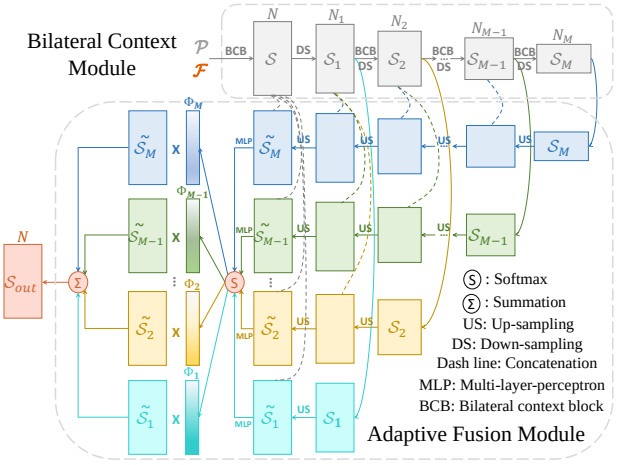


Figure 4: The architecture of the Adaptive Fusion Module. All the annotations are consistent with the items in Sec. 3 of the main paper.

respectively. In this case, the upsampled full-sized feature maps are $\{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3, \tilde{\mathcal{S}}_4, \tilde{\mathcal{S}}_5\}$, all of which are in $\mathbb{R}^{N\times 32}$.

Then, for each upsampled full-sized feature map, we use a fully-connected layer (FC, and its superscript indicates the number of kernels) to summarize the point-level information:

$$\phi_m = \mathrm{FC}^{1}(\tilde{\mathcal{S}_m}), \quad \phi_m \in \mathbb{R}^{N};$$

where $\forall\tilde{\mathcal{S}_m} \in \{\tilde{\mathcal{S}}_1, \tilde{\mathcal{S}}_2, \tilde{\mathcal{S}}_3, \tilde{\mathcal{S}}_4, \tilde{\mathcal{S}}_5\}$. Subsequently, we concatenate the $\{\phi_1, \phi_2, \phi_3, \phi_4, \phi_5\}$, and point-wisely normalize them using softmax function:

$$\Phi = \mathrm{softmax}\big(\mathrm{concat}(\phi_1, \phi_2, \phi_3, \phi_4, \phi_5)\big), \Phi \in \mathbb{R}^{N\times 5}.$$

Next, we separate Φ channel-by-channel, and obtain the fusion parameters: $\{\Phi_1, \Phi_2, \Phi_3, \Phi_4, \Phi_5\}$, all of which are in $\mathbb{R}^{N}$. Hence, the point-level adaptively fused feature map is calculated as:

$$\mathcal{S}_{out} = \Phi_1 \times \tilde{\mathcal{S}}_1 + \Phi_2 \times \tilde{\mathcal{S}}_2 + \Phi_3 \times \tilde{\mathcal{S}}_3 + \Phi_4 \times \tilde{\mathcal{S}}_4 + \Phi_5 \times \tilde{\mathcal{S}}_5,$$

where $\mathcal{S}_{out} \in \mathbb{R}^{N\times 32}$.

B.2. Predictions

Based on $\mathcal{S}_{out}$, we utilize three fully-connected layers and a drop-out layer (DP, and the drop-out ratio shows at the superscript) to predict the confidence scores for all Q candidate semantic classes:

$$\mathcal{V}_{pred} = \mathrm{FC}^{Q}\Big(\mathrm{DP}^{0.5}\big(\mathrm{FC}^{32}(\mathrm{FC}^{64}(\mathcal{S}_{out}))\big)\Big),$$

where $\mathcal{V}_{pred} \in \mathbb{R}^{N\times Q}$.

Testing Area	**mAcc**	**OA**	**mIoU**	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
Area 1	87.7	89.5	76.3	96.5	95.4	80.3	65.4	58.8	78.0	84.3	70.7	82.9	78.0	60.9	73.2	67.9
Area 2	71.1	86.6	57.8	87.1	95.1	80.0	19.8	33.3	47.5	69.3	45.6	83.1	52.8	50.7	33.1	54.4
Area 3	89.7	91.7	80.0	95.8	98.2	83.3	74.4	40.5	86.0	88.5	74.4	83.7	79.0	73.6	88.9	73.9
Area 4	77.9	86.1	64.3	94.8	97.1	78.6	53.0	48.6	30.8	61.0	67.4	77.0	70.1	51.3	44.8	61.6
Area 5	73.1	88.9	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
Area 6	92.0	92.5	81.8	96.4	97.5	86.2	79.9	81.0	78.5	90.1	77.1	88.1	65.1	72.4	79.7	71.2
6-fold	83.1	88.9	72.2	93.3	96.8	81.6	61.9	49.5	65.4	73.3	72.0	83.7	67.5	64.3	67.0	62.4

Table 7: Detailed semantic segmentation results (%) on *S3DIS* [2] dataset. (**mAcc**: average class accuracy, **OA**: overall accuracy, **mIoU**: mean Intersection-over-Union."6-fold": 6-fold cross-validation result.)

Method	**OA**	**mIoU**	man-made terrain	natural terrain	high vegetation	low vegetation	buildings	hard scape	scanning artefacts	cars
SnapNet [6]	88.6	59.1	82.0	77.3	79.7	22.9	91.1	18.4	37.3	64.4
ShellNet [53]	93.2	69.3	96.3	90.4	83.9	41.0	94.2	34.7	43.9	70.2
GACNet [**?**]	91.9	70.8	86.4	77.7	**88.5**	**60.6**	94.2	37.3	43.5	77.8
SPG [26]	94.0	73.2	**97.4**	92.6	87.9	44.0	83.2	31.0	63.5	76.2
KPConv [44]	92.9	74.6	90.9	82.2	84.2	47.9	94.9	40.0	**77.3**	**79.7**
RandLA-Net [19]	**94.8**	**77.4**	95.6	91.4	86.6	51.5	**95.7**	**51.5**	69.8	76.8
Ours	94.3	75.3	96.3	**93.7**	87.7	48.1	94.6	43.8	58.2	79.5

Table 8: Semantic segmentation (reduced-8) results (%) on *Semantic3D* [14] dataset.

B.3. Loss Function

Eq. 7 of the main paper formulates the overall loss $\mathcal{L}_{all}$ of our network based on the cross-entropy loss $\mathcal{L}_{CE}$ and the augmentation loss $\mathcal{L}_m$ for each Bilateral Context Block.

In practice, our Bilateral Context Module gradually processes a decreasing number of points ($N \rightarrow \frac{N}{4} \rightarrow \frac{N}{16} \rightarrow \frac{N}{64} \rightarrow \frac{N}{256}$) through five blocks. Empirically, we set the weights $\{0.1, 0.1, 0.3, 0.5, 0.5\}$ for the corresponding five augmentation losses, since we aim to provide more penalties for lower-resolution processing. Therefore, the overall loss for our network is:

$$\begin{aligned}\mathcal{L}_{all} =& \mathcal{L}_{CE} + \\ & 0.1 \cdot \mathcal{L}_1 + 0.1 \cdot \mathcal{L}_2 + \\ & 0.3 \cdot \mathcal{L}_3 + 0.5 \cdot \mathcal{L}_4 + 0.5 \cdot \mathcal{L}_5.\end{aligned}$$

C. Experiments

C.1. Areas of S3DIS

We include more experimental data about our network's semantic segmentation performance. To be specific, Tab. 7 shows our results for each area in the S3DIS dataset, including overall accuracy, average class accuracy, and concrete IoUs for 13 semantic classes. To evaluate each area, we apply the rest five areas as the training set.

C.2. Reduced-8 Semantic3D

Further, Tab. 8 presents our online evaluation results on the smaller test set (*i.e.*, reduced-8, which has four scenes including about 0.1 billion points) of the Semantic3D dataset. Comparing with Tab. 2 in the main paper (*i.e.*, results of semantic-8, which contains 15 scenes with 2 billion points), we conclude that our semantic segmentation performance regarding large-scale data is relatively better.

C.3. Ablation Study

In addition to the specific ablation studies (Sec. 5.3 in the main paper) about our Bilateral Context Block and Adaptive Fusion Module respectively, we also conduct an ablation study to investigate some variants of our network:

- **Baseline model:** We replace both our Bilateral Context Block and Adaptive Fusion Module with their baseline forms, which are explained in the ablation studies of the main paper.
- **Efficient model:** We apply the random sampling instead of the Farthest Point Sampling (FPS).
- **Dilated model:** We use dilated-knn [11] to search the neighbors of each point, in order to increase the size of point's receptive field. The dilated factor $d = 2$.

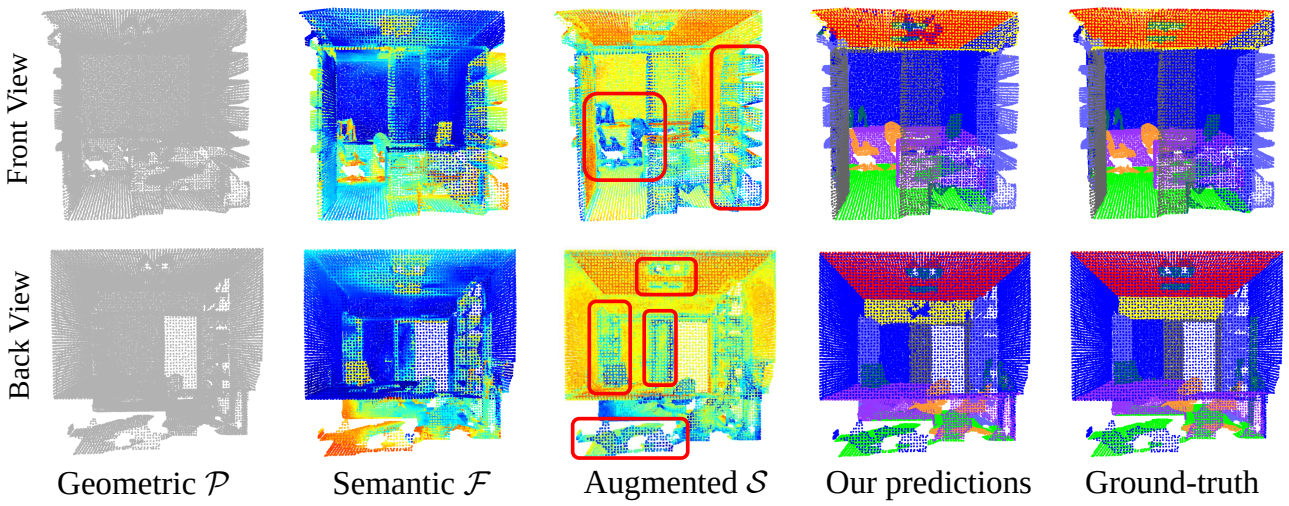


Figure 5: Visualization of intermediate features and semantic segmentation results for an office scene in *S3DIS* [2] dataset. $\mathcal{P}$ denotes the 3D coordinates of the point cloud, and $\mathcal{F}$ presents the semantic information acquired by the Feature Extractor (Sec. 4.1 in the main paper). Further, $\mathcal{S}$ means the output of our Bilateral Context Block (Sec. 3.1).

Model	Description	mIoU (%)
N_0	Baseline model	60.8
N_1	Efficient model	64.8
N_2	Dilated model	62.5
N_3	Equal-weighted model	64.0
N_4	Simplified model	63.5
N_5	**Proposed model**	**65.4**

Table 9: Ablation study about different variants of our network, tested on Area 5, *S3DIS* [2] dataset.

- **Equal-weighted model:** We set an equal weight ($\omega_i = 0.3$) for all of the augmentation losses in Eq. 7 (*i.e.*, calculating the overall loss $\mathcal{L}_{all}$) of the main paper.
- **Simplified model:** We only study four resolutions of the point cloud through the Bilateral Context Module. The number of points decreases as: $N \rightarrow \frac{N}{4} \rightarrow \frac{N}{16} \rightarrow \frac{N}{64}$, while the number of channels goes as: $16 \rightarrow 64 \rightarrow 128 \rightarrow 256$.

Tab. 9 indicates that such an efficient random sampling (N_1) cannot perform as effectively as FPS does since the randomly sampled subsets can hardly retain the integrity of inherent geometry. As there is always a trade-off between the network's efficiency and effectiveness, we look forward to better balancing them in future work. Besides, increasing the size of the point's receptive field (N_2) as [11] may not help in our case. Further, we observe that it is not optimal to use the equal-weighted Bilateral Context Blocks (N_3) for multi-resolution point clouds. Moreover, our network can be flexibly assembled: for an instance of model N_4 that consists of fewer blocks, even though the performance is reduced, it consumes less GPU memory.

Layer		1	2	3	4	5
#Points		40960	10240	2560	640	160
3D Space	**Mean**	$\downarrow 12$	$\downarrow 24$	$\downarrow 47$	$\downarrow 85$	$\downarrow 154$
	Variance	$\downarrow 0.1$	$\downarrow 0.2$	$\downarrow 0.5$	$\downarrow 2$	$\downarrow 13$
Feature Space	**Mean**	$\downarrow 45$	$\downarrow 693$	$\downarrow 814$	$\downarrow 124$	$\downarrow 317$
	Variance	$\downarrow 11.9$	$\downarrow 16.3$	$\downarrow 24.7$	$\downarrow 46.2$	$\downarrow 104$

Table 10: The general *changes* ($\times 10^{-3}$) of neighborhoods by involving bilateral offsets.

D. Visualization

D.1. Bilateral Context Block

In Fig. 5, we present the Bilateral Context Block's output features in a heat-map view. Particularly, we observe that the Bilateral Context Block can clearly raise different responses for close points (in red frames) that are in different semantic classes.

Besides, we calculate the average neighbor-to-centroid Euclidean-distances and average neighborhood variances in 3D space (Eq. 1 in the main paper) and feature space (Eq. 2), using the S3DIS samples. Tab. 10 shows that shifted neighbors get closer to centroids as expected, in both 3D and feature spaces. Further, the variances inside the neighborhoods also drop. In general, the shifted neighbors tend to form compact neighborhoods.

D.2. Visualizations and Failure Cases

We provide more visualizations of our semantic segmentation network's outputs and some failure cases. Specifically, Fig. 6 presents our results on six different types of rooms, which are *conference, WC, storage, hallway, lobby, office* rooms, respectively. Unfortunately, we find that the proposed method is not competent enough for distinguishing the objects that are in similar shapes. The main reason is that the network relies on the local neighborhood of each point, while lacks the geometric information about the specific object that each point belongs to. In the 3rd row of Fig. 6, *beam* is incorrectly classified as *door* since it looks like the doorframes; while *wall* is wrongly predicted as *board* or *clutter* in the rest of rows.

In Fig. 7, we show the general semantic segmentation performances on some large-scale point clouds of typical urban and rural scenes. Although the ground-truths of Semantic3D's test set are unavailable, our semantic predictions of these scenes are visually plausible.

In addition, we compare our results against the ground-truths on the validation set (*i.e.*, Sequence 08) of SemanticKITTI dataset in Fig. 8. Particularly, we illustrate some 3D point cloud scenes in the views of 2D panorama, in order to clearly show the failure cases (highlighted in red color). In fact, the proposed network is able to find some small objects that are semantically different from the background, however, the predictions are not accurate enough since we only use the 3D coordinates as input. As SemanticKITTI is made up of the sequences of scans, in the future, we will take the temporal information into account.

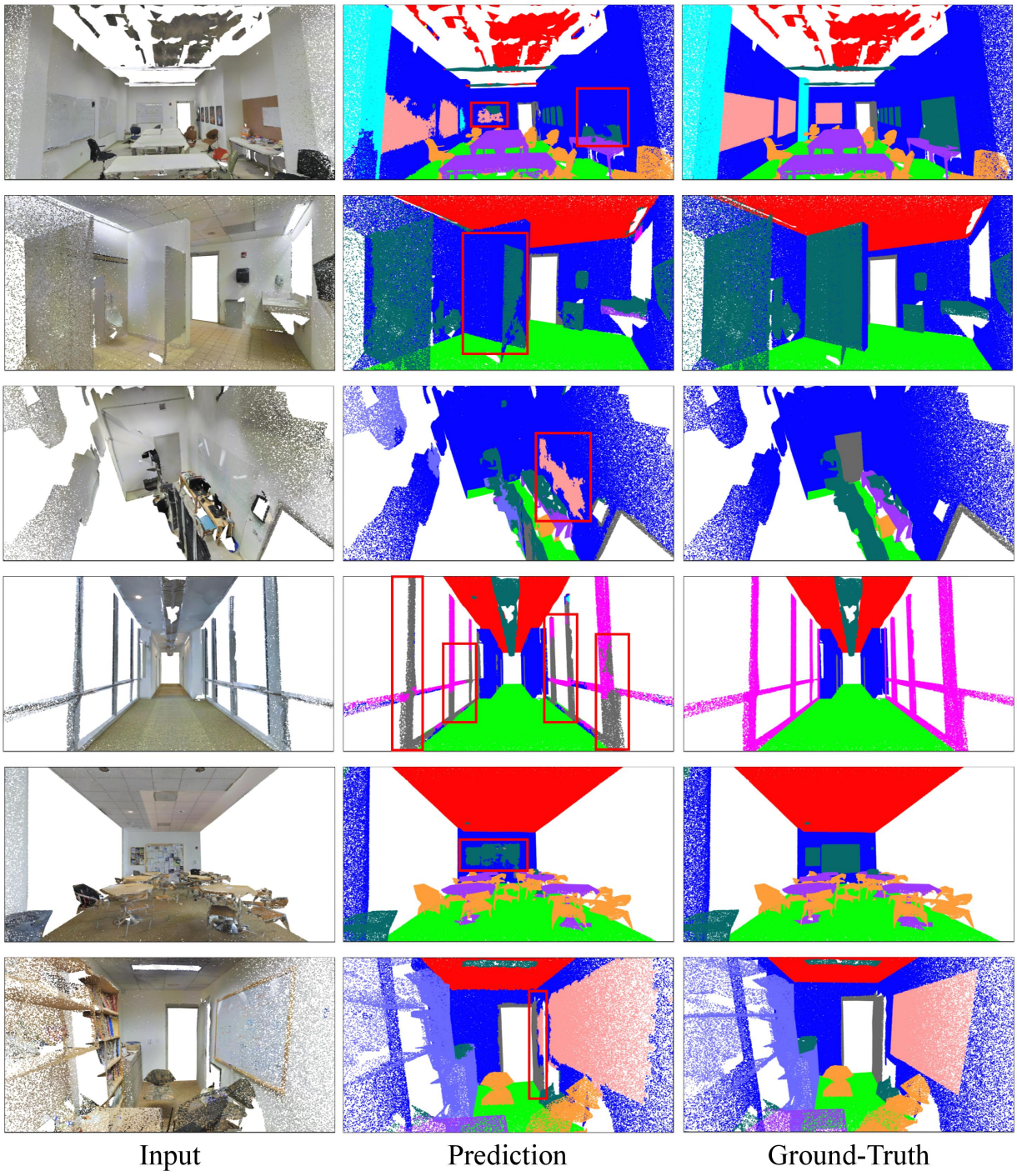


Figure 6: Examples of our semantic segmentation results of *S3DIS* [2] dataset. The first column presents the input point cloud scenes ("Input") of some indoor rooms. The second column shows the semantic segmentation predictions of our network ("Prediction"), while the last column indicates the ground-truths ("Ground-Truth"). The main differences are highlighted in red frames.

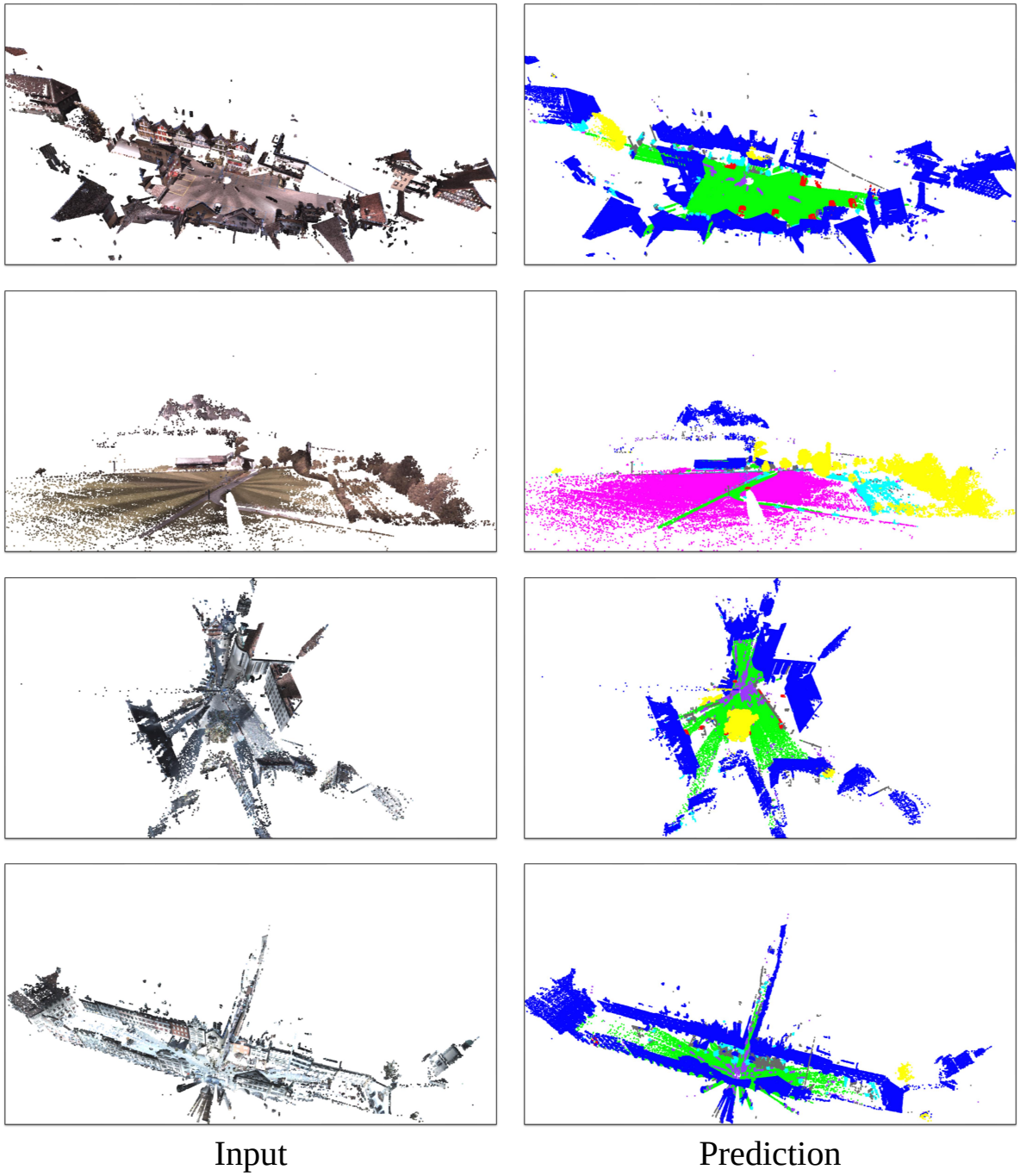


Figure 7: Examples of our semantic segmentation predictions of *Semantic3D* [14] dataset. The first row is about an urban square, the second one shows a rural farm, the third one illustrates a cathedral scene, and the last is scanned from a street view.

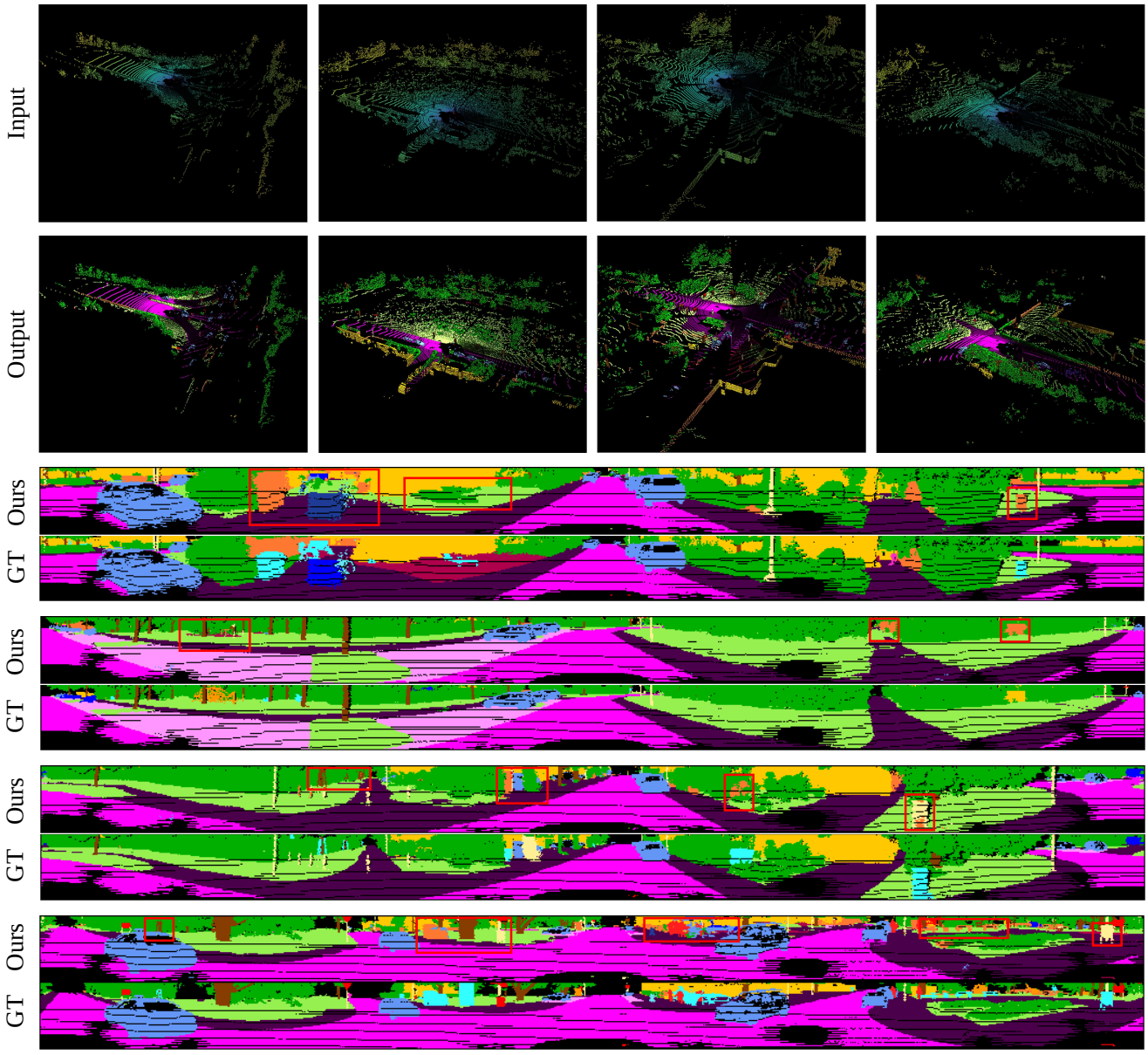


Figure 8: Examples of our semantic segmentation predictions of *SemanticKITTI* [4] dataset. The first two rows show the general 3D views of the input traffic scenarios ("Input") and our semantic segmentation outputs ("Output"), respectively. The remaining rows compare our predictions ("Ours") and the ground-truths ("GT") in 2D panorama views, where the failure cases are highlighted in red frames.