APNet: Urban-level Scene Segmentation of Aerial Images and Point Clouds

Weijie Wei Martin R. Oswald Fatemeh Karimi Nejadasl Theo Gevers
University of Amsterdam
{w.wei2, m.r.oswald, f.kariminejadasl, th.gevers}@uva.nl

Abstract

In this paper, we focus on semantic segmentation method for point clouds of urban scenes. Our fundamental concept revolves around the collaborative utilization of diverse scene representations to benefit from different context information and network architectures. To this end, the proposed network architecture, called APNet, is split into two branches: a point cloud branch and an aerial image branch which input is generated from a point cloud. To leverage the different properties of each branch, we employ a geometry-aware fusion module that is learned to combine the results of each branch. Additional separate losses for each branch avoid that one branch dominates the results, ensure the best performance for each branch individually and explicitly define the input domain of the fusion network assuring it only performs data fusion. Our experiments demonstrate that the fusion output consistently outperforms the individual network branches and that APNet achieves state-of-the-art performance of 65.2 mIoU on the SensatUrban dataset. Upon acceptance, the source code will be made accessible.

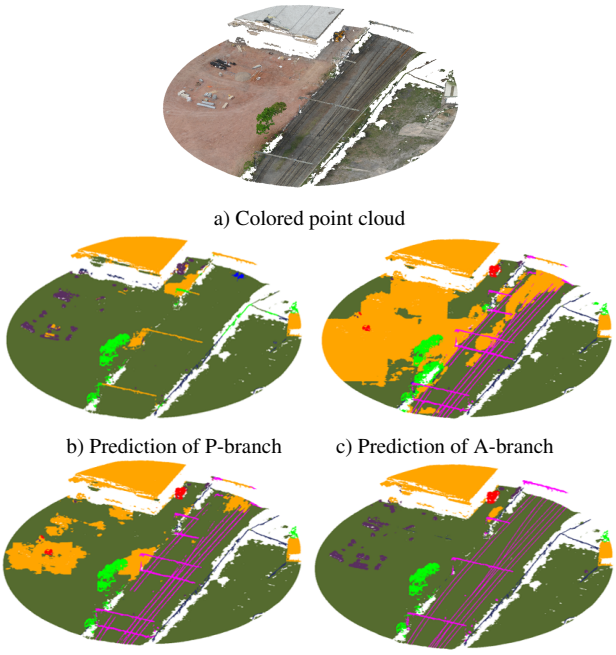
1. Introduction

Urban-level point cloud segmentation is an important stepping stone for semantic scene understanding for various applications like autonomous driving, robotics, large-scale map creation or mixed reality [15, 7]. The majority of urban semantic segmentation methods can be categorized to either use aerial / birds-eye-view image data [32, 38] or 3D point cloud data [19, 39, 10].

On the one hand, 2D/2.5D image-based approaches benefit from the simple data structure that allows for highly effective aggregation of large spatial contexts which is useful for semantic inference and for which a large pool of network architectures exist [13, 18, 32, 38]. However, these methods are limited to resolve full 3D shapes and spatial context along the gravity directions.

On the other hand, point cloud-based approaches can leverage full 3D spatial context, but context aggregation and high detail levels are generally much more expensive to progress and are thus more limited in spatial resolution and context reasoning. Unlike images, as they may suffer from large color variations due to changing weather conditions or day-to-night cycles, point clouds are more robust to these phenomena [15]. However, point clouds are more challenging to process due to their irregular and non-uniform structure. Similarly, many established network architectures exist for point cloud processing [3, 19, 26, 39, 10].

We argue that semantic reasoning in both domains has advantages and disadvantages, *e.g.* incorporating a larger context within a 2D domain enhances the recognition ca-



a) Colored point cloud
b) Prediction of P-branch
c) Prediction of A-branch
d) Prediction of APNet
e) Ground Truth

Figure 1. **APNet Segmentation.** Starting from a colored input point cloud (a) the data is fed into two separate branches: a point-cloud branch (b) and an aerial image branch (c). The key idea is to exploit the advantages of both branches regarding spatial context and network architectures. The results of both branches is then merged with a fusion network. APNet achieves a better result, which is much closer to the ground truth than the solution of individual branches.

pabilities of flat and large objects, whereas small objects with a 3D spatial extension are more effectively detectable within the 3D domain. With this objective in mind, our primary aim is to leverage and combine the best properties of both domains to propose a unified semantic segmentation approach that synergistically learns from both.

Recent papers show impressive results on vehicle-based point cloud datasets by combining different representations [16, 23, 31]. However, their corresponding representations, *e.g*. range-view and voxelization, are less suited for UAV-based datasets. To address the aforementioned objectives, we propose APNet, which concurrently operates within the aerial image domain and the point cloud domain. Exemplary results of APNet are depicted in Fig. 1. Our **contributions** can be summarized as follows:

- We introduce APNet, an effective network architecture for urban-level point cloud segmentation that leverages differences in domain properties regarding network architectures and spatial context by following a multi-branch where each branch is specialized for a particular domain.
- We propose a geometry-aware fusion module that introduces the geometric information of the original points into the process of feature fusion of two branches and achieve a better performance.
- Our experiments demonstrate the efficacy of APNet by attaining state-of-the-art performance on the SensatUrban dataset [8].

2. Related Work

2.1. Single Representation for Point Cloud Segmentation

In recent years, various deep learning-based methods are proposed for point cloud segmentation. These methods can be grouped into three categories based on their representation: projection-based, voxelization-based and point-based methods. The aim of both the projection-based and voxelization-based methods is to transform 3D point clouds to a regular representation and then use off-the-shelf networks to extract the features. In contrast, point-based methods directly process irregular point clouds.

Projection-based representation. Deep learning has made great strides in 2D computer vision tasks, leading researchers to apply the well-established 2D networks to 3D tasks. Lawin et al. [14] propose a 3D-2D-3D pipeline to solve point cloud segmentation. They project a point cloud onto multi-view 2D planes and feed the resulting images to a 2D segmentation network. The final semantic per-point label is obtained by fusing the pixel-level predictions. Although the multi-view strategy can alleviate occlusion, the pre- and postprocessing are inefficient and the results are sensitive to viewpoint selection. Furthermore, multi-view projection is typically used for a single scene or object, whereas urban-scale point clouds usually result in more occlusion. Other approaches utilize range-view planes as an intermediate representation for point cloud datasets collected by rotating laser scanner [29, 30, 18], which is a typical sensor for autonomous vehicles. In this scenery, the egocentric spherical representation can retain more information in contrast to a single plane representation. However, this representation is not well-suited for UAV-based datasets as it results in server occlusion due to the inconsistency of laser direction and projection direction. Inspired by these methods, we propose to project the point cloud onto aerial-view plane that is perpendicular to the laser. The one-time aerial-view projection is efficient and avoids information loss caused by occlusion as much as possible.

Voxelization-based representation. These methods convert a point cloud into a discrete representation, such as cubic voxels, and then use a 3D convolution neural network (CNN) to compute the features [40, 25]. This representation naturally preserves the neighborhood structure of 3D point clouds but 3D CNNs are memory and computation-intensive. These costs increase dramatically in outdoor scenarios due to the sparsity of points leading many empty voxels. Although some methods use sparse convolution to reduce these costs, the discretization unit is non-trivial to determine [23, 6]. Furthermore, urban-level datasets often contain heterogeneous objects, ranging from tiny bikes and to huge buildings and streets, which makes them unsuitable for voxelization-based methods.

Point-based representation. Point-based methods directly process irregular point clouds by different means, *e.g*. multi-layer perceptron, point convolution or graph-based operations. MLP-based networks usually stack multiple MLPs with a feature aggregation module in accordance to the convolution layers with a subsequent pooling layer in 2D neural network [3, 19, 10]. Furthermore, point convolution simulates powerful 2D convolution in 3D space by utilizing a parametric continuous convolution layer [28] or a group of kernel points as reference points [26]. Point-based methods are applicable to various datasets because they do not rely on transforming a point cloud to other intermediate representations. So far, there are only point-based methods proposed for urban-level point cloud segmentation. For instance, both EyeNet [34] and LGS-Net [21] utilize a point-based network, namely RandLA-Net [10], as their backbone. MRNet exploits multiple 3D receptive fields and LGS-Net emphasizes the utilization of geometric information. Du *et al*. [5], using KPConv [26] as the backbone, exploit a multi-task framework to achieve both boundary localization and semantic segmentation. Huang *et*

al. [11] improve a transformer-based network by applying a local context propagation module to ensure message passing among neighboring local regions. Despite numerous efforts, point-based methods remain computationally intensive. Increasing the receptive field of point-based methods is challenging, whereas this can be easily accomplished in highly-optimized 2D networks.

In summary, numerous methods have been proposed for point cloud segmentation, but a handful of them are suitable for urban-level point cloud segmentation. Additionally, single representations have their limitations. For urban-level point cloud segmentation, geometric information and large receptive fields are equally crucial. Therefore, we propose APNet to combine aerial-views and point-based representations. To the best of our knowledge, we are the first to propose a hybrid method to handle urban-level point cloud segmentation.

2.2. Hybrid Representation for Point Cloud Segmentation

There are also a number of methods that combine different representations. One common strategy is to parallelize multiple networks processing different representations and combining features at different levels. SPVNAS [23], Cylinder3D [41] and DRINet [33] share the concept of paralleling voxel-point architectures. SPVNAS [23] introduces a sparse voxel convolution and combines voxel-wise and point-wise features in different stages. Cylinder3D [41] imposes a point refinement module at the end of the network, which sums voxel-wise and point-wise features followed by three fully-connected layers. DRINet [33] introduces a voxel-point iteration module to iteratively interact between two features. RPVNet [31] consists of three branches, *i.e*. range-view, point-wise and voxel branches. A gated attention module generates coefficients for a linear combination that point-wisely combines information from three branches. These methods combine features either by a simple addition or point-wise combination but fall short to incorporate features from neighbour points. AMVNet [16] addresses this issue by training a small assertion-based network and feeding information from neighbours into it to generate final predictions. However, in the small network, only semantic predictions, *i.e*. class-wise probability scores, are considered. Hence, deeper features with richer contextual information are ignored. In conclusion, hybrid methods leverage prior knowledge from different representations to enhance features to achieve better performance. Nevertheless, the fusion module is often naive and the information from neighbour points is ignored.

Therefore, in this paper, we propose a simple yet effective fusion module that takes both contextual and geometric features as input and exploits positional relationships among neighbour points to generate descriptive features. In contrast to previous methods, our approach effectively incorporates information from neighboring points and achieves better performance on urban-level point cloud segmentation tasks.

3. Methodology

In this section, we first present the problem statement. Then, we discuss the different components of our APNet, *i.e*. the dual-encoder and the GAF. Finally, we explain the segmentation heads and define loss functions.

Problem statement. Given a colored point cloud $\mathbf{P} = \{(p_k, c_k)\}_{k=1}^{N}$ with N point coordinates $p_k = (x_k, y_k, z_k) \in \mathbb{R}^3$ and colors $c_k = (r_k, g_k, b_k) \in \mathbb{R}^3$, the aim is to compute the corresponding semantic labels $\mathbf{L} = \{(l_k)\}_{k=1}^{N}$ for every point. We train a deep learning model $h(\cdot|\theta)$ with parameter θ by minimizing the difference between the prediction $\mathbf{L} = h(\mathbf{P}|\theta)$ and corresponding ground truth label set $\hat{\mathbf{L}}$. The urban-level point cloud datasets are obtained by UAVs.

3.1. Dual-encoder

The key idea of our approach is to split up the label prediction into two different domains: an aerial (A)-branch and a point-based (P)-branch to leverage the advantages of using different spatial contexts that corresponding 2D vs. 3D network architectures have. The output of both branches is then fused within a geometry-aware fusion (GAF) module as illustrated in Fig. 2. Rather than fusing the label predictions of each branch $\mathbf{L}^a$ and $\mathbf{L}^p$, the GAF operates on intermediate feature representations $\mathbf{F}^a$ and $\mathbf{F}^p$ for a more informed label decision process. We detail both branches in the following paragraphs.

Aerial image branch. To obtain a pseudo aerial image of a point cloud, we first project it to an aerial view by an orthographic projection. Assuming that the gravity direction is aligned with the z-axis, each point $p_k = (x_k, y_k, z_k)$ is converted to a pixel $p_i = (u_i, v_i)$ via a mapping $\rho : \mathbb{R}^3 \mapsto \mathbb{R}^2$, as defined by

$$(u_i, v_i)^T = \rho(p_k) = \left(\left\lfloor \frac{x_k}{s} \right\rfloor, \left\lfloor \frac{y_k}{s} \right\rfloor \right)^T \quad , \tag{1}$$

where i is the index of a pixel and s is the quantization unit, *i.e*. pixel size. By aggregating all 3D points into pixels, we obtain the initial aerial image $\mathbf{I}^{init} \in \mathbb{R}^{H \times W \times 3}$. Note that the mapping function ρ is a many-to-one function and we only preserve the properties, *e.g*. color and label, of the highest point in the final image. Moreover, due to the sparsity of LiDAR points, a pseudo image created from the projection of a point cloud must be completed because, unlike a genuine aerial image, it contains both valid and null pixels. A pixel is considered valid if it covers a minimum

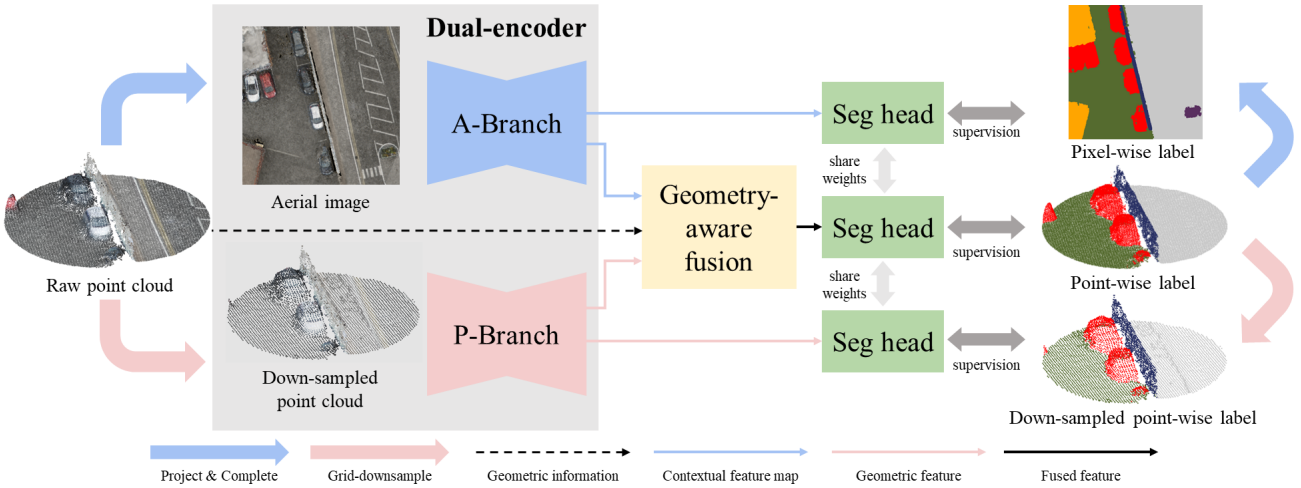


Figure 2. **Architecture overview of APNet.** The network consists of a dual-encoder, a geometry-aware fusion module and three segmentation heads that operate in different domains. The two representations of a sample, *i.e.* aerial image and down-sampled point cloud, are fed into the dual-encoder. Their outputs are passed to the fusion module for feature aggregation. Finally, the features are sent to the segmentation head for point-wise segmentation.

of one LiDAR point and is regarded as null otherwise. During the completion, valid pixels are dilated. When the eight neighbour pixels of a null pixel have more than two distinct values, its value is updated by the value that occurs most frequently among its neighbouring pixels. After the completion, we obtain the input aerial image $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$. The same projection and completion are operated on labels.

Due to the simplicity of our method, A-branch can be any end-to-end 2D semantic segmentation network. Its output is defined as follows:

$$\mathbf{F}^a = h^a(\mathbf{I}|\theta^a) \ , \tag{2}$$

where $h^a(\cdot|\theta^a)$ is the A-branch network and $\mathbf{F}^a \in \mathbb{R}^{H \times W \times C}$.

Point cloud branch. The original point cloud provides precise geometric information and is of importance in the ultimate evaluation. However, the spatial distribution of a point cloud is not uniform and local points with the same semantics tend to contain homogeneous information. To ensure the points are sampled uniformly and to increase the network's receptive field, grid-downsampling is frequently used [26, 10]. We follow KPConv [26] to perform grid-downsampling on the original point cloud, which creates a barycenter point for each non-empty grid, with the average values of all points within the same grid serving as the new properties of the barycenter point. The downsampled points are denoted as

$$\mathbf{P}^d = \{(p_k, c_k)\}_{k=1}^{N^d} \ .$$

Similar to the flexibility of the A-branch, the P-branch can be easily replaced by any point-based network and is denoted by $h^p(\cdot|\theta^p)$. By passing downsampled points to the P-branch, a point-wise feature representation is obtained:

$$\mathbf{F}^p = h^p(\mathbf{P}^d|\theta^p) \ , \tag{3}$$

where $\mathbf{F}^p \in \mathbb{R}^{N^d \times C}$.

For both the P-branch and A-branch, instead of using ultimate semantic predictions of two base models, we use the high-dimensional features from the intermediate layers of two base models.

3.2. Geometry-aware Fusion Module

In point cloud segmentation, many methods [26, 10, 31] commonly employ a preprocessing step to achieve a uniform point density. This is typically achieved through grid-downsampling, wherein the point cloud is transformed into a gird-based represetnation. During the training and validation stages, only the newly generated points are processed within the network. The postprocessing, namely upsampling, only occurs during the testing phase, where the labels of original points are determined based on the predictions of their nearest neighbouring points. However, this pipeline fails to include the features of other neighbouring points and the geometric information of the original point cloud throughout the training process. To address this, we employ a skip connection to convey geometric information of the original point cloud to the fusion module and utilize a point convolution to gather features of neighbour points. Our GAF module includes two parts, namely feature extraction and fusion, as illustrated in Fig. 3.

The feature extraction is performed at the downsampled point level to reduce the computational complexity. For a given point belonging to downsampled points $p_k^d \in \mathbf{P}^d$, its features are computed from the outputs of two branches.

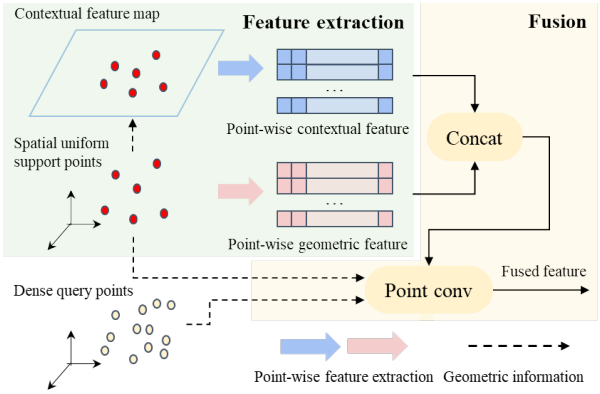


Figure 3. **Geometry-aware fusion module** includes feature extraction and fusion. Given support points and a contextual feature map, point-wise contextual features are extracted and concatenated with point-wise geometric features. The concatenated features and geometric information of both query points and support points are fed into a point convolution to aggregate geometric context information for generating the fused output features.

Specifically, the output of the P-branch, which is in the form of a point-wise feature and thus ready to use, is denoted as f_k^d for a specific point p_k^d. In the other hand, for the pixel feature, unlike the quantization operation in generating the aerial image, the bilinear interpolation and the precise 2D coordinates of the point p_k^d, *i.e.* $(u_k, v_k) = (x_k^d/s, y_k^d/s)$, are used to obtain pixel feature:

$$f_k^a = \sum_{u,v \in \delta(k)} \phi(u_k, v_k, u, v) \mathbf{F}^a(u, v) \ , \tag{4}$$

where $\delta(k)$ is the set of the four neighboring pixels of point k and $\phi(\cdot)$ computes the bilinear weights.

The process of feature involves the concatenation of features derived from the two branches and a point convolution. A point convolution, *e.g.* KPConv [26], is defined as follows,

$$f_k = \mathcal{G}(p_k) = \sum_{p_l \in \mathcal{N}_{p_k}} g(p_k - p_l) f_l \ , \tag{5}$$

where $\mathcal{G}$ represents the point convolution, while $g(\cdot)$ denotes the kernel function that computes the weights based on the vector from target point p_k to one of its neighbouring points p_l. f_l is the concatenated feature of point p_l from feature extraction module and $\mathcal{N}_{p_k}$ refers to the neighbouring points of point p_k. In summary, the feature of a target point is obtained by weighted sum the features of its neighbouring points.

For each single point convolution, we use one point from a pre-defined query set $\mathbf{P}^q$ as the target point and obtain its features based on its neighbouring points from a pre-defined support set $\mathbf{P}^s$. Note that the neighbouring point set, denoted as $\mathcal{N}_{p_k}$, is a subset of the support set $\mathbf{P}^s$. This subset is generated by considering the distances between the neighbouring points and the target point p_k. A common practice is to use a same point cloud, *e.g.* a downsampled point cloud, for both the query set and support set [26], which is denoted as the naive GAF module, as discussed in Sec. 4.3. In this way, the entire network works at the level of downsampled points. Nevertheless, our investigations indicate that the performance is negatively affected by disregarding the precise geometric information of the original points. To address this, we opt to utilise the original points $\mathbf{P}$ instead of the downsampled points $\mathbf{P}^d$ as the query set, which implies that we set $\mathbf{P}^q = \mathbf{P}$. The fused feature $\mathbf{f}_k^{\text{fused}}$ of point p_k is obtained by $\mathbf{f}_k^{\text{fused}} = \mathcal{G}(p_k)$ and the feature set is defined as $\mathbf{F}^{\text{fuse}} = \{\mathbf{f}_k^{\text{fused}} | k = 1, 2, ..N\}$.

In summary, the feature extraction operates at the level of downsampled points and the feature fusion incorporates the precise geometric information of the original points during the training stage, which enhances the accuracy.

3.3. Segmentation Heads and Loss function

The segmentation heads are a set of convolutional layers with 1×1 kernel compressing the channel from a high dimension to a low one, namely the number of categories. The final output of the model is defined by:

$$\mathbf{Pred}^{\text{rep}} = \text{Conv}_{1\times1}^{m}(\mathbf{F}^{\text{rep}}) \ , \tag{6}$$

where $\mathbf{Pred}^{\text{rep}} \in \mathbb{R}^{1 \times N_{\text{classes}}}$ is the probabilistic prediction based on the feature f^{rep} and $rep \in \{a, p, fused\}$ stands for aerial, point-wise or fused representation. $\text{Conv}_{1\times1}^{m}$ means 1×1 a convolutional layer is repeated for m times.

Two class-balanced loss function is used, *i.e.* weighted cross-entropy (WCE) with inverse frequency [4] and Lovász-softmax loss [2]. The WCE loss is applied between the output of three segmentation heads and corresponding ground-truths:

$$\mathcal{L}_1^{\text{rep}} = \mathcal{L}_{\text{WCE}}(\mathbf{Pred}^{\text{rep}}, \hat{\mathbf{L}}) \ , \tag{7}$$

Note that although three representations share the same segmentation head and the loss function, the ground-truths $\hat{\mathbf{L}}$ are different. The pixel-wise label, grid-downsampled point label and the label for raw points are applied to aerial, point-wise and fused predictions respectively. The Lovász-softmax loss is only applied to the fused representation:

$$\mathcal{L}_2 = \mathcal{L}_{\text{Lovasz}}(\mathbf{Pred}^{\text{rep}}, \hat{\mathbf{L}}) \ , \tag{8}$$

Eventually, the overall loss is calculated as:

$$\mathcal{L}_{\text{all}} = \sum_{\text{rep}=\{a,p,\text{fused}\}} \alpha^{\text{rep}} \mathcal{L}_1^{\text{rep}} + \beta \mathcal{L}_2 \ . \tag{9}$$

where α and β are the factors to adjust the scale of loss functions.

4. Experiments

In this section, we introduce the implementation details of our APNet in Sec. 4.1. Then we compare the proposed model with SOTA models on the SensatUrban dataset [8] in Sec. 4.2. Finally, the effectiveness of all components are analyzed in Sec. 4.3.

4.1. Experimental setup

SensatUrban Dataset. SensatUrban [8] is an urban-level photogrammetric point cloud dataset collected by a UAV. It covers a total of 7.64 square kilometers in three UK cities, *i.e*. Birmingham, Cambridge and York, and provides annotations for 13 semantic categories. Its average density of it is 473 points per square meter. For easier processing, the data are cut into 43 blocks with a maximum size of 400 meters by 400 meters. We follow the official split, which consists of training/validation/testing set with 33/4/6 blocks. During training and evaluation, the data from different cities are exploited mutually. We use the training set for training and report ablation studies on the validation set. We also report results on the testing set by submitting the predictions to the leaderboard where the ground truths are unpublished for a fair comparison. The grid size for down-sampling is set as 0.2 meters, resulting in 92% of the original points being filtered out. The pixel size for projection is set as 0.04 meters and the image size is set as 512×512, resulting in a coverage of $20.48m \times 20.48m$.

Metrics. As official recommendations [8], the main metric for per-category evaluation is intersection-over-union (IoU) and its mean value (mIoU) over all classes. The IoU is formulated as follows:

$$\text{IoU}_c = \frac{TP_c}{TP_c + FP_c + FN_c} \ , \tag{10}$$

where TP_c, FP_c and FN_c indicate true positive, false positive and false negative predictions for class c. The mIoU is the average IoU over all classes:

$$\text{mIoU} = \frac{1}{N_{\text{cla}}} \sum_{c=1}^{N_{\text{cla}}} \text{IoU}_c \tag{11}$$

where N_{cla} stands for the number of classes. Additionally, the overall accuracy is also reported. It is defined as follows:

$$OA = \frac{\sum_{c=1}^{N_{\text{cla}}} TP_c}{N_{\text{poitns}}} \ , \tag{12}$$

where N_{points} is the total number of points.

Implementation details. HRNet [27] with object contextual representation [35] and a variant of RandLA-Net [10] are chosen as the backbones for A-branch and P-branch, respectively. These are detailed in the supplementary materials. AdamW optimizer [17] is used with a weight decay of 0.01 and a default learning rate of 0.001, while the learning rate of the P-branch is multiplied by a factor of 5. The learning rate decreases by 5% after each epoch. The network is trained for 200 epochs for SensatUrban, with a batch size of 32. During the training procedure, random rotation along z-axis, random flip along y-axis and random scale are performed for both grid-downsampled points and aerial images while the correspondences are preserved. For more efficient training, the data in the training set and validation set are cropped into 100m $\times$ 100m patches approximately.

4.2. Comparison with existing methods

Quantitative results. The comparison of our method and other existing methods on SensatUrban benchmark [8] are shown in Table 1. Remarkably, APNet surpasses all other methods, achieving an OA of 94.0% and a mIoU of 65.2%. Notably, APNet outperforms its backbone, RandLA-Net [10], by an impressive margin of 12.5%, affirming the beneficial impact of the A-branch on segmentation. Furthermore, APNet excels in specific categories, ranking first in both the traffic road and the footpath categories. Additionally, APNet attains a top-three position in 8 out of 13 categories, further validating its superior performance.

Qualitative results. Fig. 4 is a high-level visualization to qualitatively compare the prediction of APNet and the ground truth. As indicated by the OA, APNet predicts most of the points correctly and performs excellently in the two $400m \times 400m$ blocks. Nevertheless, the primary source of inaccuracy in this figure is from the footpath, which presents problems due to its contextual and physical resemblance to the traffic road. Fig. 5 showcases a visual assessment of APNet against PushBoundary [5]. The middle column, *i.e*. the results of PushBoundary with the red dashed boxes, is taken directly from the original paper. Even though the target regions are chosen by other authors, our method shows comparable or superior performance compared to PushBoundary.

4.3. Ablation studies

Branch ablations. We first compare A-branch, P-branch and APNet. For the single branch networks, the GAF strategy is not applied as the features are obtained from single representation. The output feature from A/P-branch is directly passed to the segmentation head and generates an intermediate prediction. For A-branch, the final prediction is generated through a bilinear interpolation based on four neighbouring pixels. For P-branch, the final prediction is obtained by coping prediction from the nearest neighbour point within downsampled point set. As shown in Table 2, the combined network outperforms every single branch on

Method	OA	mIoU	ground	vegetation	building	wall	bridge	parking	rail	traffic road	street furniture	car	footpath	bike	water
PointNet [3]	80.8	23.7	67.9	89.5	80.1	0.0	0.0	3.9	0.0	31.6	0.0	35.1	0.0	0.0	0.0
PointNet++ [19]	84.3	32.9	72.5	94.2	84.8	2.7	2.1	25.8	0.0	31.5	11.4	38.8	7.1	0.0	56.9
TangentConv [24]	77.0	33.3	71.5	91.4	75.9	35.2	0.0	45.3	0.0	26.7	19.2	67.6	0.0	0.0	0.0
SPGraph [12]	85.3	37.3	69.9	94.6	88.9	32.8	12.6	15.8	15.5	30.6	22.9	56.4	0.5	0.0	44.2
SparseConv [6]	88.7	42.7	74.1	97.9	94.2	63.3	7.5	24.2	0.0	30.1	34.0	74.4	0.0	0.0	54.8
KPConv [26]	93.2	57.6	**87.1**	**98.9**	95.3	74.4	28.7	41.4	0.0	55.9	54.4	85.7	40.4	0.0	**86.3**
RandLA-Net [10]	89.8	52.7	80.1	98.1	91.6	48.9	40.6	51.6	0.0	56.7	33.2	80.1	32.6	0.0	71.3
BAF-LAC [22]	91.5	54.1	84.4	98.4	94.1	57.2	27.6	42.5	15.0	51.6	39.5	78.1	40.1	0.0	75.2
BAAF-Net [20]	92.0	57.3	84.2	98.3	94.0	55.2	48.9	57.7	20.0	57.3	39.3	79.3	40.7	0.0	70.1
LGS-Net [21]	93.3	63.6	86.1	98.7	95.7	65.7	**62.8**	52.6	36.5	62.0	52.1	84.3	45.9	9.0	75.0
PushBoundary [5]	93.8	59.7	85.8	**98.9**	**96.8**	**79.3**	49.7	52.4	0.0	62.1	**57.6**	**86.8**	42.0	0.0	65.5
LCPFormer [11]	93.5	63.4	86.5	98.3	96.0	55.8	57.0	50.6	**46.3**	61.4	51.5	85.2	49.2	0.0	86.2
LACV-Net* [37]	93.2	61.3	85.5	98.4	95.6	61.9	58.6	64.0	28.5	62.8	45.4	81.9	42.4	4.8	67.7
EyeNet [34]	93.7	62.3	86.6	98.6	96.2	65.8	59.2	**64.8**	17.9	64.8	49.8	83.1	46.2	**11.1**	65.4
U-Next* [36]	93.0	62.8	85.2	98.6	95.0	68.2	53.6	60.4	36.8	64.0	48.9	84.9	45.1	0.0	76.2
APNet (Ours)	**94.0**	**65.2**	86.7	98.3	95.8	75.2	49.7	60.5	42.6	**66.3**	52.6	85.1	**50.9**	1.2	82.6

Table 1. **Comparison with SOTA methods on SensatUrban online benchmark [8]**. Our method performs often better on rare classes which are difficult to label in one or the other domain. * indicates arXiv paper. Best results are highlighted as **first**, second, and third.

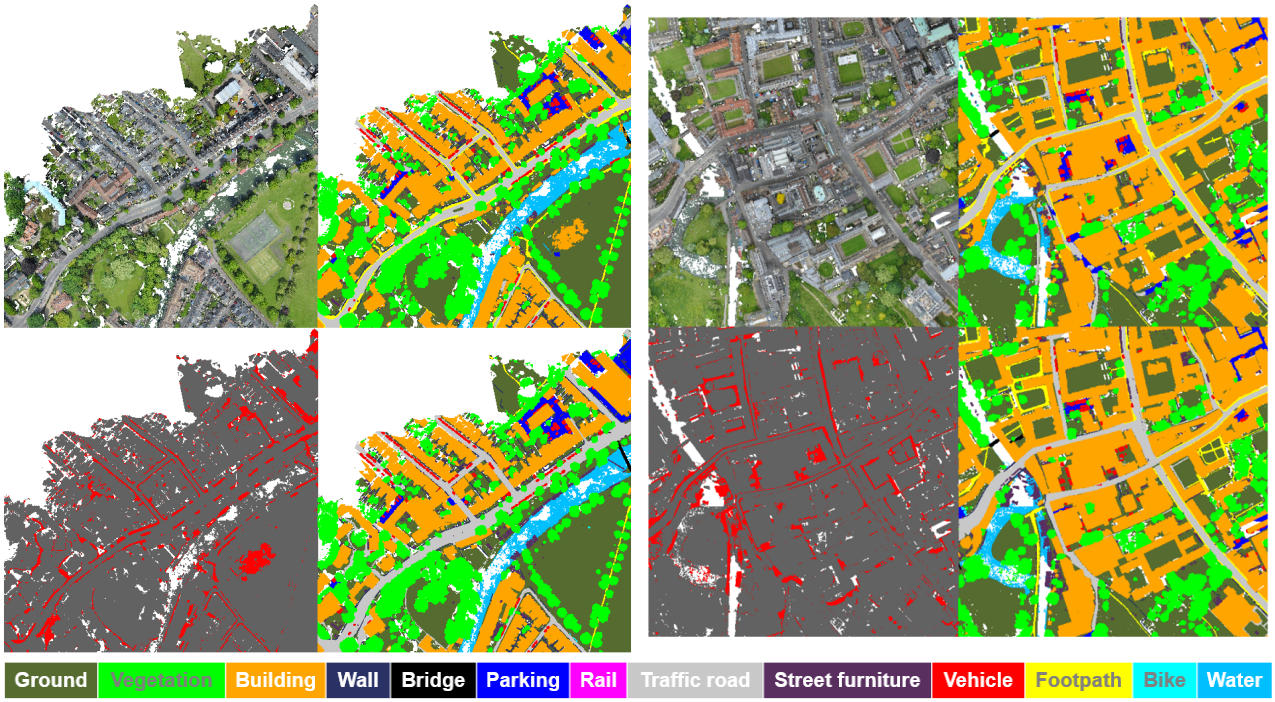


Figure 4. **The qualitative result of two blocks in the validation set of SensatUrban [9].** In each figure, the top-left sub-figure is the aerial visualization of the original point cloud which covers a $400m \times 400m$ area. The top-right and bottom-right sub-figures are the predictions and ground truth respectively. Both of them follow the color bar at the bottom. The bottom-left sub-figure is an error map that presents the difference between the prediction and the ground truth.

OA, mIoU and most of the IoUs of all categories. In cases where APNet performs worse than single-branch networks, the difference is negligible. Notably, P-branch outperforms A-branch on OA, although the opposite is observed for most categories. This is because of the imbalanced distribution of categories in the dataset. Over 50% of the points are attributed to the three categories - ground, vegetation, and building, resulting in that a higher accuracy for these dominant categories will mask shortcomings in other categories for an overall metric.

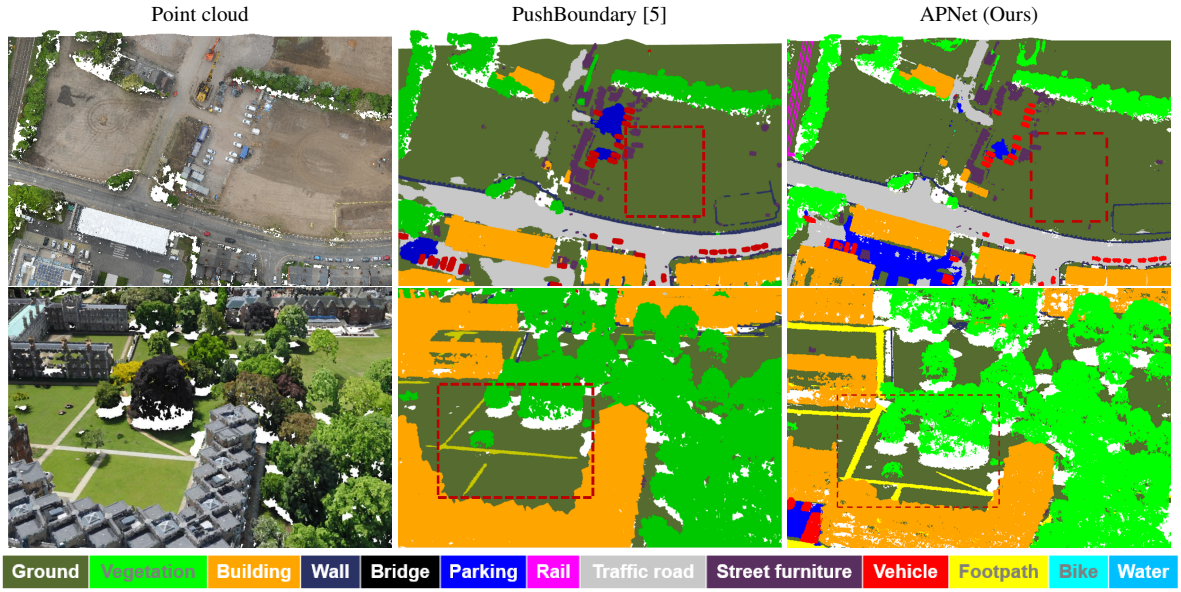


Figure 5. **Qualitative comparison with PushBoundary [5] on the SensatUrban [8] test set (No GT available).** The figures of the PushBoundary with the red dash boxes are directly taken from the original paper. APNet performs on par with PushBoundary in the first example (top row) and outperforms it in the second example (bottom row).

Method	OA	mIoU	ground	vegetation	building	wall	bridge	parking	rail	traffic road	street furniture	car	footpath	bike	water
A-branch	89.8	55.2	71.8	91.6	94.3	70.0	22.9	47.2	**46.6**	65.4	45.4	81.1	**20.4**	0.0	61.0
P-branch	90.2	52.1	75.0	95.4	93.3	52.4	**27.4**	40.7	23.3	59.3	34.3	80.6	18.2	**12.4**	65.2
APNet (Ours)	**92.3**	**59.2**	**80.5**	**97.4**	**96.7**	**73.0**	21.8	**52.3**	43.4	**66.1**	**50.7**	**84.8**	19.9	12.3	**70.9**

Table 2. **Ablation studies on branches.** This table compares the semantic labeling performance of the aerial image branch and the point cloud branch against the output of the geometry-aware fusion module. The benefit of the fusion module is apparent as is mostly yields better class-wise performances then the individual branches separately.

Encoder	Fusion strategy	OA	mIoU
A-branch	N/A	89.8	55.2
P-branch	N/A	90.2	52.1
Dual-encoder	Addition	91.3	56.7
	Concatenation	90.7	56.7
	Naive GAF	91.5	57.5
	GAF	**92.3**	**59.2**

Table 3. **Ablation studies on geometry-aware fusion (GAF) module.** Compared to the simpler point-wise fusion approaches (addition, concatenation), the geometry-aware fusion includes spatial context into the reasoning yielding improved performance.

Fusion strategy. We compare GAF module with two simple fusion strategies and the naive version of GAF in Table 3. The addition is the most intuitive way to combine two features. The concatenation increases the complexity slightly because a subsequent MLP is necessary to reduce the number of channels. These two combinations are point-wise and thus no neighbouring features are considered. Nevertheless, they outperform both single-branch networks. Naive GAF enhance its local adaptive capabilities by involving neighbour features at a downsampled points level. The proposed GAF improves the naive GAF by using the original points as query points and achieves the best performance on both OA and mIoU. Our GAF module yields enhanced outcomes, surpassing the simple fusion strategy by 1% OA and 2.5% mIoU, *e.g.* addition and concatenation. Ablation studies illustrate the effectiveness and necessity of each component in the proposed method.

5. Conclusion

We presented a semantic segmentation method that exploits the advantages of both point cloud-based and aerial image-based methods in a single network architecture with two separate domain branches. The reasoning about which branch is more effective for which class category and spatial location is learned by a geometry-aware fusion network that combines the output of both branches into a single estimate. Ablation studies and comparisons to state-of-the-art methods show clear benefits of the proposed architecture.

6. Acknowledgements

This work is financially supported by TomTom, the University of Amsterdam and the allowance of Top consortia for Knowledge and Innovation (TKIs) from the Netherlands Ministry of Economic Affairs and Climate Policy. Fatemeh Karimi Nejadasl is financed by the University of Amsterdam Data Science Centre.

A. Implementation details

A-branch. We adopt HRNet [27] with object-contextual representations (OCR) [35], denoted as HRNet-OCR, as the backbone for A-branch. During training, the OCR loss is preserved while the original 2D segmentation head is removed. The intermediate features, also known as augmented representations as defined in the original paper, from HRNet-OCR are compressed to a total of 128 channels, thereby ensuring alignment with the output of the P-branch.

P-branch. We employ RandLA-Net [10] as the backbone for the P-branch and follow its official configuration for the SemanticKITTI dataset[1] with the following two modifications: Firstly, we double all feature channels in the RandLA-Net to accommodate the additional color features. Furthermore, we double the output channel for the last layer to ensure compatibility with the A-branch. Consequently, the encoder produces outputs with channel dimensions of 64, 128, 256, and 512, respectively. Secondly, we input the same point cloud to RandLA-Net twice and sum up the output features. Although the network does not change, the down-sampling within the network is random, leading to different features for the same point cloud in the end. This technique promotes the consistency of RandLA-Net.

GAF module. We adopt KPConv [26] as the point convolution in the GAF module and adhere to the configuration of the rigid KPConv. Accordingly, one single rigid KPConv encompasses a sphere with a radius of 0.5 meters, centered at the query point. Each kernel point exerts an influence on all support points within a sphere whose radius is 0.24 meters and centered on the kernel point.

References

[1] 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 *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9296–9306. IEEE, 2019.

[2] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4413–4421, 2018.

[3] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. PointNet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 77–85. IEEE, 2017.

[4] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving. *arXiv preprint arXiv:2003.03653*, 3(7), 2020.

[5] Shenglan Du, Nail Ibrahimli, Jantien Stoter, Julian Kooij, and Liangliang Nan. Push-the-boundary: Boundary-aware feature propagation for semantic segmentation of 3d point clouds. *2022 International Conference on 3D Vision (3DV)*, 2022.

[6] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. *CVPR*, 2018.

[7] 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*, 43(12):4338–4364, 2020.

[8] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. Towards semantic segmentation of urban-scale 3d point clouds: A dataset, benchmarks and challenges. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[9] Qingyong Hu, Bo Yang, Sheikh Khalid, Wen Xiao, Niki Trigoni, and Andrew Markham. SensatUrban: Learning semantics from urban-scale photogrammetric point clouds. *International Journal of Computer Vision*, 130(2):316–343, 2022.

[10] 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 *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11105–11114. IEEE, 2020.

[11] Zhuoxu Huang, Zhiyou Zhao, Banghuai Li, and Jungong Han. Lcpformer: Towards effective 3d point cloud analysis via local context propagation in transformers. *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[12] 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 (CVPR)*, June 2018.

[13] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019.

[14] 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.

[15] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Michael A. Chapman, Dongpu Cao, and Jonathan Li. Deep learning for LiDAR point clouds in autonomous driving: A review. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2020.

[16] Venice Erin Liong, Thi Ngoc Tho Nguyen, Sergi Widjaja, Dhananjai Sharma, and Zhuang Jie Chong. AMVNet: Assertion-based multi-view fusion network for LiDAR semantic segmentation. *arXiv:2012.04934 [cs]*, 2020.

[17] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[18] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. RangeNet ++: Fast and accurate LiDAR semantic segmentation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4220. IEEE, 2019.

[19] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[20] 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.

[21] Yuyuan Shao, Guofeng Tong, and Hao Peng. Mining local geometric structure for large-scale 3d point clouds semantic segmentation. *Neurocomputing*, 500:191–202, 2022.

[22] Hui Shuai, Xiang Xu, and Qingshan Liu. Backward attentive fusing network with local aggregation classifier for 3d point cloud semantic segmentation. *IEEE Transactions on Image Processing*, 30:4973–4984, 2021.

[23] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European Conference on Computer Vision*, 2020.

[24] 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.

[25] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. SEGCloud: Semantic segmentation of 3d point clouds. In *International Conference on 3D Vision (3DV)*, 2017.

[26] Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, Francois Goulette, and Leonidas Guibas. KPConv: Flexible and deformable convolution for point clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6410–6419, 2019.

[27] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019.

[28] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[29] Bichen Wu, Alvin Wan, Xiangyu Yue, and Kurt Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *ICRA*, 2018.

[30] 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 *ICRA*, 2019.

[31] Jianyun Xu, Ruixiang Zhang, Jian Dou, Yushi Zhu, Jie Sun, and Shiliang Pu. RPVNet: A deep and efficient range-point-voxel fusion network for LiDAR point cloud segmentation. In *ICCV*, pages 16024–16033, 10 2021.

[32] Zhishuang Yang, Wanshou Jiang, Bo Xu, Quansheng Zhu, San Jiang, and Wei Huang. A convolutional neural network-based 3d semantic labeling method for als point clouds. *Remote Sensing*, 9(9):936, 2017.

[33] Maosheng Ye, Shuangjie Xu, Tongyi Cao, and Qifeng Chen. DRINet: A dual-representation iterative learning network for point cloud segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7447–7456, October 2021.

[34] Sunghwan Yoo, Yeonjeong Jeong, Maryam Jameela, and Gunho Sohn. Human vision based 3d point cloud semantic segmentation of large-scale outdoor scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 6576–6585, June 2023.

[35] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. In *ECCV*, 2020.

[36] Ziyin Zeng, Qingyong Hu, Zhong Xie, Jian Zhou, and Yongyang Xu. Small but mighty: Enhancing 3d point clouds semantic segmentation with u-next framework. *arXiv preprint arXiv:2304.00749*, 2023.

[37] Ziyin Zeng, Yongyang Xu, Zhong Xie, Wei Tang, Jie Wan, and Weichao Wu. Lacv-net: Semantic segmentation of large-scale point cloud scene via local adaptive and comprehensive vlad. *arXiv preprint arXiv:2210.05870*, 2022.

[38] Ruibin Zhao, Mingyong Pang, and Jidong Wang. Classifying airborne lidar point clouds via deep features learned by a multi-scale convolutional neural network. *International journal of geographical information science*, 32(5):960–979, 2018.

[39] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1009–1018, 2019.

[40] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[41] Xinge Zhu, Hui Zhou, Tai Wang, Fangzhou Hong, Yuexin Ma, Wei Li, Hongsheng Li, and Dahua Lin. Cylindrical and

asymmetrical 3d convolution networks for LiDAR segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9939–9948, 6 2021.