Deep FusionNet for Point Cloud Semantic Segmentation

Feihu Zhang[1(✉)], Jin Fang[2], Benjamin Wah[3], and Philip Torr[1]

[1] University of Oxford, Oxford, UK
feihu.zhang@eng.ox.ac.uk
[2] Baidu Research, Beijing, China
[3] Chinese University of Hong Kong, Sha Tin, Hong Kong S.A.R.

Abstract. Many point cloud segmentation methods rely on transferring irregular points into a voxel-based regular representation. Although voxel-based convolutions are useful for feature aggregation, they produce ambiguous or wrong predictions if a voxel contains points from different classes. Other approaches (such as PointNets and point-wise convolutions) can take irregular points for feature learning. But their high memory and computational costs (such as for neighborhood search and ball-querying) limit their ability and accuracy for large-scale point cloud processing. To address these issues, we propose a deep fusion network architecture (FusionNet) with a unique voxel-based "mini-PointNet" point cloud representation and a new feature aggregation module (fusion module) for large-scale 3D semantic segmentation. Our FusionNet can learn more accurate point-wise predictions when compared to voxel-based convolutional networks. It can realize more effective feature aggregations with lower memory and computational complexity for large-scale point cloud segmentation when compared to the popular point-wise convolutions. Our experimental results show that FusionNet can take more than one million points on one GPU for training to achieve state-of-the-art accuracy on large-scale Semantic KITTI benchmark.The code will be available at https://github.com/feihuzhang/LiDARSeg.

1 Introduction

Semantic segmentation of 3D Point cloud is widely applicable in many scenarios, including remote sensing, AR/VR, robotics, and self-driving cars. Many deep neural network models have been proposed for this important task.

Approaches typically rely on a voxel-based regular representation that converts unordered points to regular 3D voxel/grids before using 3D/2D convolutions for feature learning [11,23,29,36,36,43,54,54]. Inspired by the success in image-based segmentation, these volumetric networks can be efficiently designed and trained for semantic segmentation of 3D point clouds, using regular convolutions which have been shown useful for coarse-grain feature aggregation/learning

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58586-0_38) contains supplementary material, which is available to authorized users.

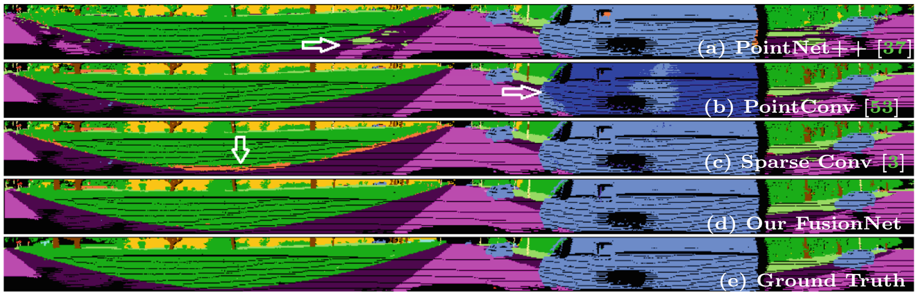


Fig. 1. Problem illustrations with large-scale point clouds (Semantic KITTI [2]). The LiDAR frame contains more than 120K points in a large 3D space of 160 m × 160 m × 20 m. Results are projected into 2D cylindrical images for visual comparisons. *Top rows (a) and (b):* State-of-the-art PointNet++ [37] and point-wise convolutions [53] fail in large objects/areas (*e.g.* sidewalk, car) due to insufficient feature propagations/aggregations for large-scale point clouds. *Third row (c):* State-of-the-art sparse convolutions [3] predict wrong labels at the border between different classes due to their ambiguous and coarse voxel-level learning. *Fourth row (d):* Our FusionNet gives accurate segmentation labels for the large-scale point cloud. *Last row (e):* ground truth.

[3,7,61]. However, they can only give voxel-level predictions. Moreover, their voxelization process transfers many raw points to one single voxel that will produce ambiguous or wrong predictions at object borders when voxels consist of points from different classes (as illustrated in Fig. 1(c)).

When compared to regular convolution operations, PointNets [35,37] can take irregular points for feature learning. However, the Multi-Layer-Perceptron (MLP) [35] in PointNets keeps only the most significant activation and may lose some useful detailed information for segmentation. Also, these models are limited for training deep and robust networks for large-scale point clouds (Fig. 1(a)) because they have high memory and computational complexity in their neighborhood search, sampling, and ball-querying operations [35,37,50,57].

There is also plenty of work on point-based convolutions that explores the idea of convolutions directly on irregular points [8,25,26,46,47,55]. They learn to approximate a weight function or interpolate convolutional weights [10,25,28,32, 50,53,55]. Compared to volumetric convolutions that can directly index a kernel with fixed relative positions of the neighborhoods, the positions of neighbors become unpredictable in these point-wise convolutions as the points are scattered irregularly. Hence, the kernel for neighboring points must be calculated on the fly. The additional memory costs and matrix multiplications will limit the training of effective networks for large-scale point clouds and may produce wrong predictions in some objects due to insufficient feature aggregations (Fig. 1(b)).

To address the challenges of designing effective network architectures for accurate point-wise segmentation of large-scale point clouds, we develop a deep fusion network architecture with a unique voxel-based "mini-PointNet" structure for point cloud representation and a novel fusion module (Fig. 2) for feature aggregation. The proposed FusionNet realizes both the voxel aggregation and the fine-grain point-wise feature learning. It inherits all the advantages (in terms of effectiveness and efficiency) of volumetric convolutional networks (*e.g.*, 3D UNet [41]) while being able to learn point-wise features for accurate label predictions.

FusionNet possesses many advantages over existing models in large-scale point cloud segmentation:

1) When compared to existing voxel networks [3,7,61], FusionNet can predict point-wise labels and avoid those ambiguous/wrong predictions when a voxel has points from different classes.
2) When compared to the popular PointNets [35,37,50,57] and point-based convolutions [25,53], FusionNet has more effective feature aggregation operations (including the efficient neighborhood-voxel aggregations and the fine-grain inner-voxel point-level aggregations). These operations help produce better accuracy for large-scale point cloud segmentation.
3) FusionNet takes full advantage of the sparsity property to reduce its memory footprint. For instance, it can take more than one million points in training and use only one GPU to achieve state-of-the-art accuracy.

With an effective feature aggregation module and lower memory and computation costs, we can train our FusionNet to learn more effective deep features for accurate large-scale point cloud segmentation (as illustrated in Fig. 1d). In our experiments, FusionNet achieves state-of-the-art performance on large-scale point cloud datasets. Especially, it outperforms state-of-the-art PointNets [37] (by 40%), point-wise convolutions [53] (by 10%) and sparse CNN [3] (by 7%) in mean IoU evaluations on the challenging Semantic KITTI benchmark.

2 Related Work

Existing approaches for 3D point cloud segmentation can be roughly categorized into two types: regular voxel-based networks and irregular point-based networks.

2.1 Voxel-Based Networks

There is substantial previous work on 3D CNN to convert point clouds to 2D or 3D volumetric grids/voxels (or similar slices/lattices) [29,36,54,61]. For example, 3D point clouds or shapes have been projected into several 2D image-like grids with 2D convolutions for shape classification and retrieval tasks [36,43]. Other studies [11,23,29,36,54] voxelize point clouds into 3D volumetric voxels and apply 3D convolution networks for point cloud processing and understanding.

Among these studies, VV-Net [30] uses a kernel-based interpolated variational auto encoder (VAE) on regular voxel grids; instance segmentation models have been proposed on dense 3D voxels/2D grids [19,61]; panoptic labels can be predicted using a spatially hashed volumetric map [31]; high-resolution RGB inputs have been leveraged by associating 2D images with the volumetric grid [11]; efficient feature aggregations have been explored in PVCNN [27] through a voxel-based regular CNN in low resolution to avoid random memory accesses.

To extend voxel-based networks to high-resolution scene segmentation tasks [2,4], a set of unbalanced octrees have been used to improve the resolution [40]. Sparse Submanifold CNN [3,7] computes the convolutions only at activated points to design high-resolution volumetric inputs. However, higher resolution inputs mean more sparse inputs, making convolutional layers less efficient for feature aggregation. Also, it will increase memory and computation costs.

2.2 Point-Based Networks

Recent work takes raw points as input for feature learning and label predictions.

PointNets: These methods aggregate features from different points by shared multi-layer perceptrons (MLP) [35]. PointNet++ improves its network by adding a hierarchical structure [37]. Wang *et al.* associate instances and semantics segmentation [49] with feature encoder of stacked PointNet layers. He *et al.* learn deep geodesic-aware representations for PointNet/PointNet++ [9]. The issue with the MLP module is that they keep only the most significant activation on features, leading to the loss of some useful detailed information for segmentation.

Point-Wise Convolutions: Other recent work explores the extension of convolutions on irregular and unordered point-cloud inputs [8,25,26,46,47,55].

For unordered points, PointCNN performs convolutions on transformed points [25]. Zhang *et al.* use statistics from concentric spherical shells to resolve point-order ambiguities [62]. FeaStNet generalizes conventional convolution layers by adding a soft-assignment matrix [47]. KPConv learns flexible and deformable point convolutions [46]. Point-wise Conv [13,53], GeoCNN [20] and annular convolution [18] query the nearest neighbors for convolutions with different kernels.

Some work explores contiguous convolutions [32,50,53] for point clouds. For instance, PointConv [53] treats convolution kernels as nonlinear functions of the coordinates and uses (MLP) modules to learn continuous weight functions for point-wise convolutions. Hermosilla *et al.* develop Monte Carlo convolutions for learning non-uniformly sampled point clouds [10]. Similarly, Mao *et al.* propose to interpolate discrete convolutional weights for convolutions on points [28].

Other work explores the ideas of convolutions on unique surfaces. Rao *et al.* map 3D points onto a discretized sphere for convolution definition [38]. Huang *et al.* utilize a 4-rotational symmetric field to define a domain for convolution on a surface [14]. Tangent convolution projects local surface geometry on a tangent plane around every point, producing planar convolution-able tangent images [44].

In the original regular 2D/3D convolutions, the relative positions of the neighborhoods are fixed, and the convolutional kernel can be directly indexed. However, for point-based convolutions, the points are irregularly scattered over a 3D space, making the relative positions of neighbors unpredictable. Hence, the kernel for each neighboring point must be calculated on the fly using additional matrix multiplications.

There are also graph or tree-based convolutions for point clouds, including local spectral graph convolutions [48], graph attention convolutions [49], regular-

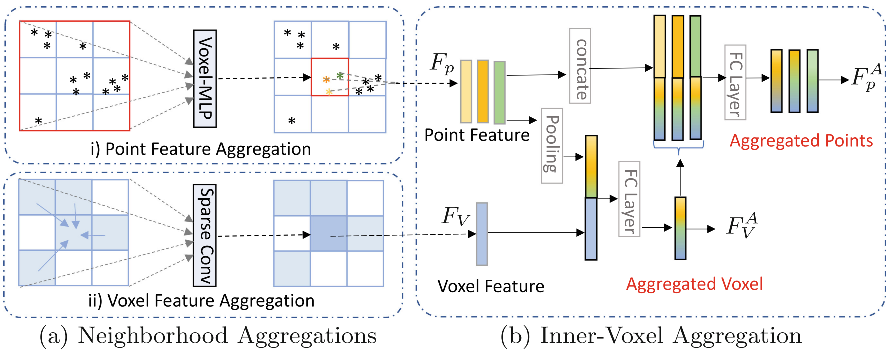


Fig. 2. Illustartion of the FusionNet module. (a) Neighborhood aggregation, including i) point-level feature aggregation by the proposed Voxel-MLP and the ii) voxel-level feature aggregation with sparse convolutions. (b) Inner-voxel aggregation of each voxel in a feedback fashion. Features of all the points in the "mini-PointNet" are fused into voxel-level features by average-pooling and concatenation. The voxel-level features are then fed back to each point (with concatenation and refinement).

ized graph CNN (RGCNN) [45], and octree guided CNN with spherical kernels [24]. The unique feature of them is that they often rely on a graph construction and a learned kernel function for convolutions on unordered points. These are realized by additional modules/layers with extra computations and memory overhead which limit their abilities for designing effective deep architectures in large-scale point cloud processing.

Other Point Networks: There are also special networks developed for various applications to directly take raw points as input for sampling [58,60], semantic [17,39,51,59,63,64], and instance segmentation [33,57]. Some employ new and special data structures for point cloud processing, including 3D point capsule encoder and decoder networks [65], over segmentation on graph structure [21], basis point sets [34], multi-resolution tree-structured networks [5], bilateral convolutions on a sparse lattice of the point cloud [42], recurrent neural networks (RNN) on slices of a point cloud [15], and the superpoint graph-based semantic segmentation [22].

3 FusionNet

Many previous models are either limited by the ambiguous voxel-level predictions [7,29,61] or have insufficient feature aggregations that use high memory or computational costs [25,27,37,61] in large-scale point-cloud segmentation. This section describes our deep FusionNet model that can learn accurate point-wise features and realize more effective and memory-efficient feature aggregations in large-scale processing. It utilizes a unique voxel-based "mini-PointNet" for sparse representation and the novel FusionNet modules for feature aggregations.

Figure 2 illustrates the FusionNet module that includes both **neighborhood voxel aggregation** (Fig. 2(a)) for voxel-level learning and **inner-voxel aggregation** for fine-grain point-level learning (Fig. 2(b)). The rest of this section is organized as follows. Section 3.1 describes the unique point cloud representation. Section 3.2 and 3.3 present the design of the FusionNet module (Fig. 2). Section 3.4 shows the up- and down-sampling layers. Finally, Sect. 3.5 describes the architecture and its sparse implementation.

3.1 Point Cloud Representation

Our point cloud representation is based on a unique voxel-based "mini-PointNet" that has two steps: voxelization and "mini-PointNet" construction.

Voxelization: Given point cloud P in a range of $L_x \times L_y \times L_z$ as input, we transfer the irregular points into regular 3D voxels with a resolution of $H \times W \times D$. The resolution is controlled by the voxel parameter $s = (s_x, s_y, s_z)$ (length/width/height of the each voxel). Here, $(H, W, D) = (L_x/s_x, L_y/s_y, L_z/s_z)$.

Mini-PointNet: Each point p can be represented as $p = \{(p_x, p_y, p_z), F_p\}$ with (p_x, p_y, p_z) as its 3D space location and F_p as its point features (*e.g.* color, intensity). Since many voxels may have more than one point, voxel V with m points can be represented as a "mini-PointNet": $V = \{(V_x, V_y, V_z), F_V, \{p_1, p_2, ...p_m\}\}$, where (V_x, V_y, V_z) is the coordinate of the voxel that contains m points $\{p_1, p_2...p_m\}$, and F_V is the voxel features that can be learned from those points in each voxel.

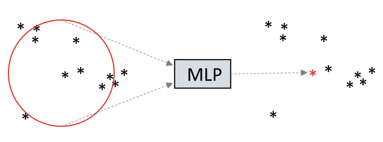
The voxel-based "mini-PointNets" are stored in sparse data structures to reduce their memory requirement. Empty or invalid voxels are not be stored or processed during feature aggregations.

After transferring the point cloud into the voxel-based "mini-PointNet" representation, our FusionNets use the proposed feature aggregation modules to learn deep feature representation for segmentation.

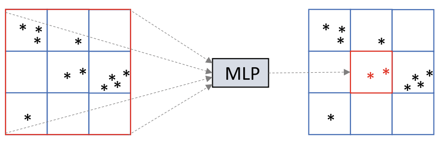
3.2 Neighborhood Aggregations

As illustrated in Fig. 2(a), there are two types of feature aggregations can propagate information from neighborhood voxels to the current voxel: voxel feature aggregation and point feature aggregation.

Voxel Feature Aggregation: We utilize regular convolutions for voxel feature aggregation, as the convolutional layer has been proven to be successful in 2D-image semantic segmentation. The knowledge can be easily transferred to the design and training of deep network models for 3D point clouds. Regular voxel-based convolutions are also more efficient than many existing point-wise convolution models [10,25,28,32,50,53,55]. These models requires extra memory and computations to learn or interpolate the weight kernels of the point



(a) MLP in PointNet



(b) Our Voxel-MLP

Fig. 3. (a) The original MLP module widely used in PointNets [35,37] and point-wise convolutions [13,53]. It relies on neighborhood search/ball querying for each pixel with high time complexity ($O(n^2)$ or $O(n \log(n))$ with k-d tree). (b) Our voxel-level MLP (*e.g.* kernel size = 3) directly aggregates features from all the points in the neighborhood voxels to points in the target voxel, allowing neighborhoods to be easily visited in regular voxels. The Voxel-MLP has a lower $O(n)$ time complexity.

Algorithm 1: Implementation of the Voxel-MLP

```
for each voxel $V$: do
    for all points $p$ ($p \in N(V)$) in neighboor voxels $N(V)$ of $V$ do
        Spatial encoding:
          i) spatial coordinate shifts $\Delta p \leftarrow (p_x, p_y, p_z) - (V_x, V_y, V_z)$;
          ii) concatenation and FC layer: $\mathbf{F}'_p \leftarrow \text{fc}(\text{cat}\{\mathbf{F}_p, \Delta p\})$;
        Point feature max-pooling: $\mathbf{F}_{max} \leftarrow \max\{\mathbf{F}'_p, p \in N(V)\}$;
    end
    for each point $p$ in current voxel $V$ do
        Concatenate $\mathbf{F}'_p$ and $\mathbf{F}_{max}$: $\mathbf{F}''_p \leftarrow \text{cat}(\mathbf{F}'_p, \mathbf{F}_{max})$;
        Refinement with point-wise fully-connected layer: $\mathbf{F}^A_p \leftarrow \text{fc}(\mathbf{F}''_p)$;
    end
end
Result: Aggregated point feature $\mathbf{F}^A_p$ ($p \in V$)
```

convolutions, as well as to locate the neighborhood points from irregular point cloud by KNN search or ball querying.

To reduce the memory footprint and to develop deep neural networks for the segmentation of large-scale point clouds, we use the Sparse Submanifold Convolution layer [3,7] for voxel-based aggregation to compute the convolution only at activated voxels (as illustrated in step ii) of Fig. 2(a)). This approach helps minimize the memory needed.

Point Feature Aggregation: Figure 3(b) presents a more efficient voxel-based multi-layer perceptron (Voxel-MLP) for neighborhood feature aggregation.

For the m points $\{p_1, p_2, ...p_m\}$ in the current voxel V, their point feature can be aggregated from all the points in the neighborhood voxels $N(V)$ (*e.g.* 27 neighborhood voxels if the kernel size is 3) by our Voxel-MLP as follows:

$$\text{mlp}(p_1, p_2, ...p_m) = \gamma \left(\max_{p_k \in N(V)} \{h(p_k)\}, \{p_1, p_2, ...p_m\} \right). \tag{1}$$

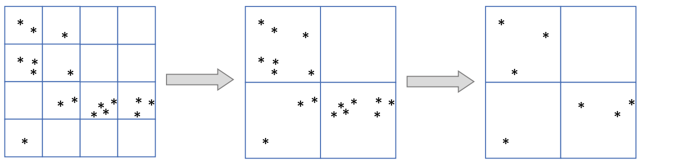


Fig. 4. Illustration of feature down-sampling. (a) Voxels and "mini-PointNets" inputs. (b) After down-sampling with stirde of 2, the size of the voxels is doubled to get a lower resolution. (c) The number of points in each voxel is also reduced to realize the "mini-PointNet" down-sampling.

The implementation details are presented in Algorithm 1. The response of h can be interpreted as the spatial encoding [35] of a point. The feature aggregation γ can be realized by concatenations and fully connected layers.

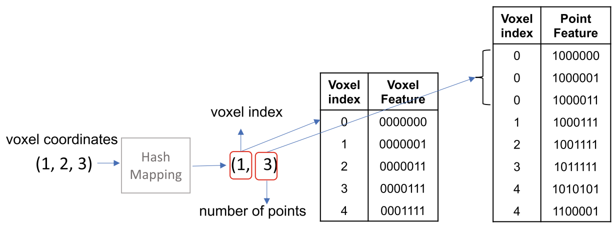
Our Voxel-MLP (Fig. 3(b)) is similar to the original MLP module (Fig. 3(a)) widely used in PointNets [35,37] and point-wise convolutions [13,53] for segmentation. However, the ball querying, sampling, or KNN neighborhood search in these existing models need $O(n^2)$ time complexity for points selection in irregular point clouds. Although k-d trees can be used to accelerate the search with $O(n \log(n))$ complexity, building a k-d tree for each layer will also cost extra memory and computations. In contrast, in our Voxel-MLP, all the neighborhood points can be directly visited more efficiently from neighborhood voxels (by hash mapping with $O(1)$ time complexity for each point–in total $O(n)$).

The receptive field of the Voxel-MLP is controlled by the kernel size that is the same as our convolutional aggregation. For example, when setting the kernel size to 3, neighborhood points will be selected from 27 neighborhood voxels.

3.3 Inner-Voxel Aggregation

Voxel-level aggregations only learn coarse-grain voxel features that usually produce ambiguous/wrong predictions at the object borders when voxels consist of points from different classes. To address this issue, we design the inner-voxel aggregation for fine-grain point-level feature aggregations and label predictions.

Figure 2(b) illustrates the inner-voxel aggregation realized by a feedback design of the feature fusion. The point-wise feature F_p and voxel feature F_V are from the outputs of the neighborhood aggregation block (Fig. 2(a)). They are sent to the inner-voxel block for fusion and aggregation. We first use point-wise average pooling to achieve the pooled feature vector from m points in the "mini-PointNet." The feature vector is then fused into the voxel feature F_V by concatenation. After refinement by one 1×1 convolution/FC layer, the aggregated voxel-level features F_V^A are then fed back to each point in the "mini-PointNet" by concatenations. Finally, a point-wise fully connected layer is employed to get the aggregated point feature F_p^A.



Voxel index	Voxel Feature
0	0000000
1	0000001
2	0000011
3	0000111
4	0001111

Voxel index	Point Feature
0	1000000
0	1000001
0	1000011
1	1000111
2	1001111
3	1011111
4	1010101
4	1100001

Fig. 5. Illustration of the sparse data structure and the voxel/point visiting by hash mapping.

Namely, for each valid voxel V with m points $(\{p_1, p_k ... p_m\} \in V)$ in the "mini-PointNet", the inner-voxel aggregation can be represented as:

$$\begin{aligned} F_V^A &= \gamma_1 \left(\underset{p \in V}{\text{avg}} \{h(F_p, p))\}, F_V \right) \\ F_p^A &= \gamma_2 \left(h(F_p, p), F_V^A \right), \ p \in V. \end{aligned} \tag{2}$$

Where the response of h can be interpreted as the spatial encoding of the point p. It can be learned by fully-connected layers with the point feature F_p and its spatial shifts to the voxel center $(p_x - V_x, p_y - V_y, p_z - V_z)$ as inputs (which is similar to [35] and the Voxel-MLP). γ_1 and γ_2 are the feature fusion functions which are realized by concatenations and fully-connected layer.

After the circulation of the inner-voxel aggregation, the voxel-level features and the point features are deeply fused. They are then sent to the next FusionNet module/layer for further aggregations.

3.4 Down/Up-Sampling

Down- and up-sampling are essential in neural networks for segmentation since they help capture pyramidal features in different resolutions and receptive fields.

For regular voxels, we use convolutions and transposed convolutions with strides (*e.g.* 2) for down- and up-sampling. These are similar to existing image segmentation models (*e.g.* UNet [41]) that have been found to be effective.

Figure 4 illustrates our point feature down-sampling that strictly follows voxel-level sampling. After voxel-level down-sampling, the size of a voxel is expanded to get a lower-resolution representation. To realize point-level down-sampling, we reduce the number of points in each voxel by re-sampling to get the new "mini-PointNet". For example, we reduce the number of points of each voxel to half while ensuring at least one valid point to avoid new empty voxels (Fig. 4(c)).

When compared to the slow iterative farthest-sampling strategy [37] (which sometimes takes 1–2 s for large-scale point cloud), the point sampling in FusionNet is realized in each voxel independently and can be computed in parallel by GPU in super fast speed (10–100 times faster than the farthest-sampling strategy). Moreover, voxel-based sampling guarantees each valid voxel to have at least one point and avoids the appearance of new empty voxels. This is especially important for sparse point cloud or regions (such as LiDAR point cloud where points are very sparse in the distance). Our voxel-based down-sampling can keep the consistency between point-level and voxel-level networks that cannot be realized by random sampling and the farthest-sampling strategy.

For point up-sampling, we use point interpolations proposed in PointNet [35] to recover the high-resolution point representations.

3.5 Architecture and Sparse Implementation

Architecture:[1] We use the 3D UNet backbone which is the same as that in [3,7]. At each pyramid level, one FusionNet module is employed to replace the convolutional layer. Both point-level and voxel-level features are densely connected to the previous layers by concatenations.

Sparse Implementation: Figure 5 illustrates the storage of the points and voxels in a sparse data structure. Each voxel and its points can be visited quickly by hash mapping. The coordinates (including the batch ID) of the voxels are used as the keys to the hash map, and a querying coordinate directly points to the location of each voxel and the "mini-PointNet".

When compared to dense regular convolution nets [46,61], FusionNet takes full advantage of sparsity to minimize its memory requirement. It differs from PointNets [35,37] and point-wise convolutional nets [13,53] that need high computational complexity for neighborhood search. As FusionNet utilizes a regular-voxel representation and hash mapping to realize the $O(1)$ point/voxel indexing, it is more suitable for large-scale point cloud segmentation tasks.

4 Experiments

In our experiments, we train our model with conventional cross-entropy loss for 40k iterations on one GPU. We use Momentum SGD with the Poly scheduler to train networks from learning rate 1e-1 and apply data augmentation including rotation around the gravity axis, scaling, spatial translation and spatial elastic distortion. For evaluations, we use the standard mean Intersection over Union (mIoU) and mean Accuracy (mAcc) as metrics following the previous works.

4.1 Datasets

We use three point cloud datasets for evaluations. Two of them are collected from the indoor scenes, and the rest one is from the large-scale driving scenes.

S3DIS: Stanford 3D Indoor Spaces (S3DIS) dataset [1] contains scans of six floors of three buildings. We use the Fold #1 split following many previous works.

[1] Architecture details are in the supplementary materials: www.feihuzhang.com.

ScanNet: The ScanNet [4] 3D semantic segmentation benchmark consists of 3D reconstructions of real rooms which has 1.5k rooms and 20 classes for semantic segmentation. Each point cloud scan contains 7–550k points.

Semantic KITTI: The Semantic KITTI dataset is a new large-scale LiDAR point cloud dataset in driving scenes. It has 22 sequences with 19 valid classes, and each scan contains 10–13k points in a large space of $160\,\text{m} \times 160\,\text{m} \times 20\,\text{m}$. We use Sequences 0–7 and 9–10 as the training set (in total 19k frames), Sequence 8 (4k frames) as the validation set, and the remaining 11 sequences (20k frames) as the test set. Different from other point cloud datasets, LiDAR points are distributed irregularly in a large 3D space. There are many small objects with only a few points and the points are very sparse in the distance. All these make it challenging for semantic segmentation of the large-scale LiDAR point clouds.

4.2 Ablation Study

We test the performance of the models with different feature aggregation settings: 1) only using the regular convolutional aggregation like that in [3,7]; 2) with both the convolutional voxel-feature aggregation and the neighborhood-voxel point feature aggregation; and 3) with all three types of feature aggregations (our FusionNet).

Table 1 shows that only with the regular voxel-based convolutional aggregation, it will become a sparse convolutional nets [3,7] and achieves 54.8% in the mean IoU evaluation. By introducing the neighborhood point feature aggregation, the results can be improved by 4%. On the contrary, the FusionNet will degenerate to pointnets [37] if we only use the point feature aggregation. Finally, the full settings of the FusionNet get the best mean IoU of 63.7%.

4.3 Benchmark Evaluations

1) Semantic KITTI: We use a voxel size of 5 cm for building our voxel-based "mini-PointNet" representation. Each valid "mini-PointNet" contains 1–60 points. The model is trained on the training set, and the results are submitted to the online benchmark for evaluation using the 20k test set.

The results are presented in Table 2 and visualized in Fig. 6 for comparisons. Our FusionNet can achieve a new state-of-the-art performance in both the mean IoU and the accuracy evaluations. It outperforms the PointNet++ [37] (by 40%), state-of-the-art point-wise convolutions [53] (by 10%) and the sparse convolution [3] (by 7%) in mean IoU evaluations. Moreover, It achieves the best IoUs in 14 out of the 19 classes. For many small objects (*e.g.* person, bicycle, motorcycle), it outperforms other existing models by at least 10–25%.

FusionNet has many advantages for the large-scale LiDAR point cloud segmentation. Compared to state-of-the-art voxel-based networks [3,7], FusionNet can predict point-wise labels and avoid those ambiguous/wrong predictions at object boundaries when a voxel has points from different classes. When compared to state-of-the-art point-wise convolutions (*e.g.* [50]), our FusionNet gets

Table 1. Ablation study on large-scale Semantic KITTI dataset.

Neighborhood-voxel aggregation		Inner-voxel aggregation	Mean accuracy (%)	Mean IoU
Voxel feature agg	Point feature agg			
√			89.6	54.8
	√	√	86.2	41.6
√	√		90.7	58.9
√	√	√	91.8	63.7

Table 2. Single scan results (19 classes) on Semantic KITTI benchmarks. All models are trained on the training set and evaluated on the online benchmark.

Approach	**mIoU**	**mAcc**	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic sign
PointNet [35]	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
SPGraph [22]	17.4	-	45.0	28.5	0.6	0.6	64.3	49.3	0.1	0.2	0.2	0.8	48.9	27.2	24.6	0.3	2.7	0.1	20.8	15.9	0.8
SPLATNet [42]	18.4	-	64.6	39.1	0.4	0.0	58.3	58.2	0.0	0.0	0.0	0.0	71.1	9.9	19.3	0.0	0.0	0.0	23.1	5.6	0.0
PointNet++ [37]	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
SqzeSegV2[52]	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
TanConv[44]	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[56]	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
DarkNet53[4]	49.9	87,8	**91.8**	74.6	64.8	27.9	84.1	86.4	25.5	24.5	32.7	22.6	78.3	50.1	64.0	36.2	33.6	4.7	55.0	38.9	52.2
RandLANet[12]	50.3	85.9	88.0	67.9	56.9	15.5	81.1	94.0	**42.7**	19.8	21.4	**38.7**	78.3	60.3	59.0	47.5	48.8	4.6	49.7	44.2	38.1
PointConv[53]	51.2	87.1	88.9	68.4	58.9	19.7	84.6	93.1	37.8	20.7	22.9	38.1	79.9	62.8	60.7	46.2	39.1	5.5	52.0	48.1	44.7
MinkNet42[3]	54.3	89.5	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	**61.3**	**91.2**	**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**

much better segmentation accuracy in the large-scale LiDAR dataset. This is because our FusionNet is realized with more effective feature aggregation operations (including the effective voxel-level neighborhood aggregations and the fine-grain inner-voxel point-level aggregations).

2) 3DSIS and ScanNet: We also evaluate our FusionNet on two indoor-scene datasets (3DSIS [1] and ScanNet[4]). The results are presented in Table 3 and Table 4. The voxel size is set to 5cm for our voxel-based "mini-PointNet" representation. When compared with the sparse convolutional nets [3] which use the same resolution as input, our FusionNet model can learn fine-grain point-wise features and give point-wise predictions. Therefore, it can get better mean IoUs (68.8% on ScanNet benchmark and 67.2% on S3DIS test set). Also, our FusionNet outperforms state-of-the-art point-wise convolutions (*e.g.* PointCNN [25], PointConv [53]) by 2–10% in the mean IoU evaluations. This is because our FusionNet is realized with a more powerful feature aggregation module for learning more effective features in the segmentation.

4.4 Analysis of the Voxel-Based "Mini-PointNet"

Our FusionNet is realized with the unique voxel-based "mini-PointNet" representation. The voxel size is the key parameter that decides the resolution of our "mini-PointNet" representation and will influence the accuracy of the segmentation results. We evaluate the performance by using different voxel sizes for

Table 3. Stanford Area 5 Test (Fold #1) (S3DIS) [1]

Methods	mIoU	mAcc
PointNet [35]	41.09	48.98
SparseUNet [6]	41.72	64.62
PVConv [27]	52.3	–
TangentConv [44]	52.80	60.70
3D RNN [59]	53.40	71.30
PointCNN[25]	57.26	63.86
SuperpointGraph [22]	58.04	66.50
MinkNet32 [3]	65.35	71.71
KPConv[46]	67.1	**72.8**
Our FusionNet	**67.2**	72.3

Table 4. 3D Semantic Label Benchmark on ScanNet [3].

Methods	mIoU
ScanNet [4]	30.6
PointNet++ [37]	33.9
TangetConv [44]	43.8
SurfaceConv [32]	44.2
MVPNet [16]	64.1
PointConv [53]	66.6
PointASNL [56]	66.6
MinkNet42 (5cm) [3]	67.9
KPConv [46]	68.4
FusionNet (5cm)	**68.8**

building the networks. Similar networks (PVCNN [27] and sparse convolutions networks [3]) are used for comparisons. They use either the dense voxel representation [27] or the sparse voxel representation [3]. Models are tested on the Semantic KITTI validation set. We fixed the number of points to 125k for the memory test (in the training phase).

As shown in Fig. 7, higher resolution can help get better accuracy. But, the memory and computational costs significantly increase (almost cubically) which is a big limitation for learning on large-scale point clouds. State-of-the-art voxel-based convolutional nets [3,7] require an extremely high resolution to achieve state-of-the-art accuracy. By using the same resolution setting, our FusionNet can get far better accuracy. The FusionNet with a lower resolution (5 cm) outperforms the high-resolution (3 cm) sparse convolutional network [3]. This is benefited from our unique voxel-based "mini-PointNet" representation that can help learn point-wise predictions with more effective feature aggregations.

4.5 Efficiency for Large-Scale Point Cloud Processing

In this section, we systematically evaluate the efficiency and abilities of our FusionNet on processing real-world large-scale point clouds. The Semantic KITTI dataset [2] is used for evaluations. We compare our FusionNet with the recent work that can also produce point-wise predictions. For fair comparisons, we fixed the number of points (to 120k) for each LiDAR scan during the speed and memory test. In addition, we also measure the maximum number of 3D points each network can take in a single pass to infer per-point semantics. All experiments are conducted on the same device (RTX TITAN GPU).

Due to the high memory and computational costs and the limitation of the architecture design, many state-of-the-art point-wise segmentation models [12, 35,37,53] require re-sampling to reduce and fix the number of points in each point

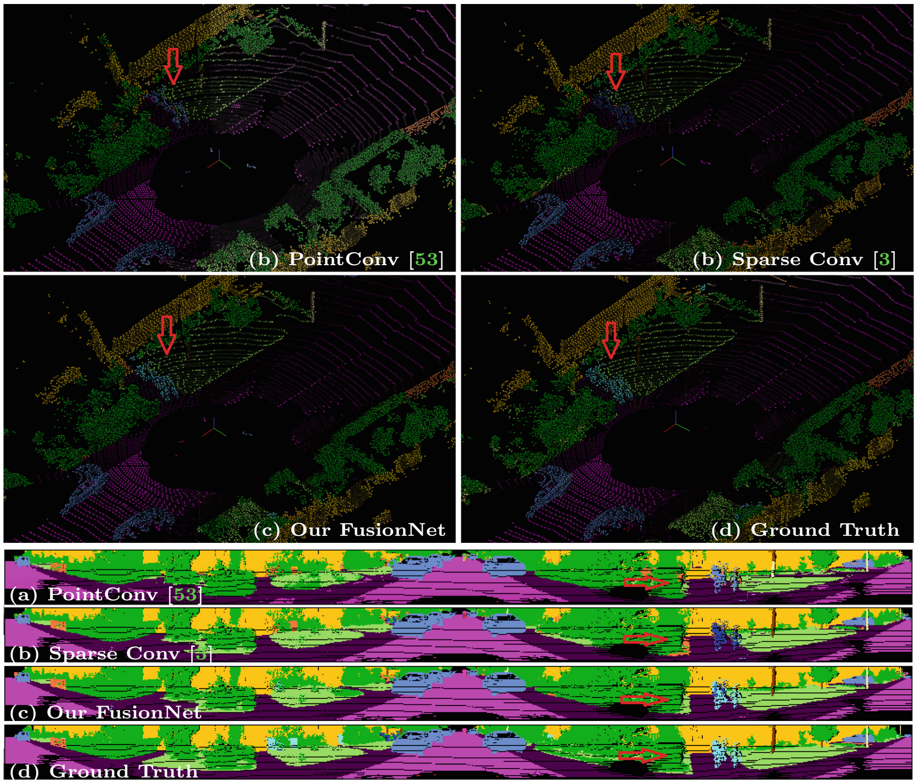
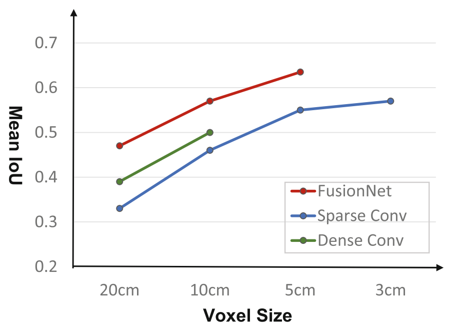


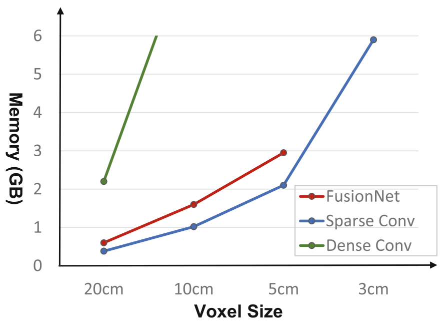
Fig. 6. Visualization of the results of LiDAR point clouds. Top (a-d) are the raw point cloud results. Bottom (a-d) are the visual comparisons by projecting the point clouds to cylindrical images. (a) State-of-the-art point-wise convolutions [53], (b) state-of-the-art sparse convolutions [3], (c) our FusionNet, (d) ground truths. The differences are as illustrated by red arrows (where bicycles are predicted as other objects by [3,53]).

cloud frame for training and testing. This need to split the point cloud. But, in LiDAR point clouds, points are very sparse in the distance, down-sampling will further increase the sparsity and lead to worse segmentation accuracy due to the insufficient feature aggregation from neighborhoods.

As a comparison, our FusionNet can directly take the entire point cloud as input for training and testing that is more flexible in real applications. As shown in Table 5, it can take up to 2.5 million points in a single pass to infer the point-wise segmentation which is four times as many as that of PointNet++.



(a) IoU Changing Trends



(b) Memory Changing Trends

Fig. 7. Performance illustration with different voxel sizes. (a) The mean IoU changes along with the voxel sizes. PVCNN [27] and sparse convolutions [3]) are used for comparisons. Higher resolution can help get a better mean IoU. (b) The memory consumption changes along with the voxel sizes (training phase using LiDAR frame [2]). The memory consumption significantly increases as the resolution goes up. Our FusionNet with a lower resolution (5 cm) can achieve a better mean IoU when compared to the sparse convolutions [3] (with a high resolution of 3 cm).

Table 5. Efficiency comparisons for large-scale point cloud processing.

Approach	Memory consumption (GB)	Elapsed time (per scan)	Max inference points (million)
PointNet [35]	3.1	0.2	0.8
PointNet++ (MSG) [37]	4.2	2.0	0.6
PointCNN [25]	13.4	3.0	0.2
PointConv [53]	3.0	3.0	0.8
Our FusionNet	1.0	0.9	2.3

5 Conclusions and Future Work

In this paper, we propose a deep fusion network architecture (FusionNet) with a new feature aggregation module for large-scale 3D semantic segmentation. The proposed FusionNet utilizes a unique voxel-based "mini-PointNet" for point cloud representation and learning. It can realize both the neighborhood voxel-level feature aggregation and the fine-grain point-wise feature learning. Therefore, FusionNet can produce more accurate point-wise predictions when compared to voxel-based convolutional networks. Also, when compared to popular point-wise convolutions, it realizes more effective feature aggregations with lower memory and computational costs.

Currently, part of our system (the hash mapping) is implemented on the CPU that limits the running speed. We will try the GPU hash mapping in the future that will significantly accelerate our FusionNet for real-time applications.

Acknowledgement. Research is supported by Baidu, the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC grant Seebibyte EP/M013774/1 and EPSRC/MURI grant EP/N019474/1. We would also like to acknowledge the Royal Academy of Engineering and FiveAI.

References

1. Armeni, I., et al.: 3D semantic parsing of large-scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1534–1543 (2016)
2. Behley, J., et al.: Semantickitti: a dataset for semantic scene understanding of lidar sequences. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 9297–9307 (2019)
3. Choy, C., Gwak, J., Savarese, S.: 4D spatio-temporal convnets: minkowski convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3075–3084 (2019)
4. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: richly-annotated 3D reconstructions of indoor scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5828–5839 (2017)
5. Gadelha, M., Wang, R., Maji, S.: Multiresolution tree networks for 3D point cloud processing. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 105–122. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_7
6. Graham, B.: Sparse 3D convolutional neural networks. arXiv preprint arXiv:1505.02890 (2015)
7. Graham, B., Engelcke, M., van der Maaten, L.: 3D semantic segmentation with submanifold sparse convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
8. Groh, F., Wieschollek, P., Lensch, H.P.A.: Flex-convolution. In: Jawahar, C.V., Li, H., Mori, G., Schindler, K. (eds.) ACCV 2018. LNCS, vol. 11361, pp. 105–122. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-20887-5_7
9. He, T., Huang, H., Yi, L., Zhou, Y., Wu, C., Wang, J., Soatto, S.: Geonet: deep geodesic networks for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
10. Hermosilla, P., Ritschel, T., Vázquez, P.P., Vinacua, À., Ropinski, T.: Monte Carlo convolution for learning on non-uniformly sampled point clouds. ACM Trans. Graph. (TOG) **37**(6), 1–12 (2018)
11. Hou, J., Dai, A., Nießner, M.: 3D-SIS: 3D semantic instance segmentation of RGB-D scans. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4421–4430 (2019)
12. Hu, Q., et al.: Randla-net: efficient semantic segmentation of large-scale point clouds. arXiv preprint arXiv:1911.11236 (2019)
13. Hua, B.S., Tran, M.K., Yeung, S.K.: Pointwise convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 984–993 (2018)
14. Huang, J., Zhang, H., Yi, L., Funkhouser, T., Nießner, M., Guibas, L.J.: Texturenet: consistent local parametrizations for learning from high-resolution signals on meshes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4440–4449 (2019)

15. Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3D segmentation of point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
16. Jaritz, M., Gu, J., Su, H.: Multi-view pointnet for 3D scene understanding. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (2019)
17. Jiang, L., Zhao, H., Liu, S., Shen, X., Fu, C.W., Jia, J.: Hierarchical point-edge interaction network for point cloud semantic segmentation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
18. Komarichev, A., Zhong, Z., Hua, J.: A-CNN: annularly convolutional neural networks on point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
19. Lahoud, J., Ghanem, B., Pollefeys, M., Oswald, M.R.: 3D instance segmentation via multi-task metric learning. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 9256–9266 (2019)
20. Lan, S., Yu, R., Yu, G., Davis, L.S.: Modeling local geometric structure of 3D point clouds using geo-CNN. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
21. Landrieu, L., Boussaha, M.: Point cloud oversegmentation with graph-structured deep metric learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
22. Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
23. Le, T., Duan, Y.: Pointgrid: a deep network for 3D shape understanding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
24. Lei, H., Akhtar, N., Mian, A.: Octree guided CNN with spherical kernels for 3D point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
25. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: PointCNN: convolution on x-transformed points. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 820–830 (2018)
26. Liu, Y., Fan, B., Meng, G., Lu, J., Xiang, S., Pan, C.: Densepoint: learning densely contextual representation for efficient point cloud processing. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
27. Liu, Z., Tang, H., Lin, Y., Han, S.: Point-voxel CNN for efficient 3D deep learning. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 963–973 (2019)
28. Mao, J., Wang, X., Li, H.: Interpolated convolutional networks for 3D point cloud understanding. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
29. Maturana, D., Scherer, S.: Voxnet: a 3D convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 922–928. IEEE (2015)
30. Meng, H.Y., Gao, L., Lai, Y.K., Manocha, D.: VV-Net: voxel VAE net with group convolutions for point cloud segmentation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
31. Narita, G., Seno, T., Ishikawa, T., Kaji, Y.: Panopticfusion: online volumetric semantic mapping at the level of stuff and things. arXiv preprint arXiv:1903.01177 (2019)

32. Pan, H., Liu, S., Liu, Y., Tong, X.: Convolutional neural networks on 3D surfaces using parallel frames. arXiv preprint arXiv:1808.04952 (2018)
33. Pham, Q.H., Nguyen, T., Hua, B.S., Roig, G., Yeung, S.K.: JSIS3D: joint semantic-instance segmentation of 3D point clouds with multi-task pointwise networks and multi-value conditional random fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
34. Prokudin, S., Lassner, C., Romero, J.: Efficient learning on point clouds with basis point sets. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
35. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 652–660 (2017)
36. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view CNNs for object classification on 3D data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5648–5656 (2016)
37. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 5099–5108 (2017)
38. Rao, Y., Lu, J., Zhou, J.: Spherical fractal convolutional neural networks for point cloud recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
39. Rethage, D., Wald, J., Sturm, J., Navab, N., Tombari, F.: Fully-convolutional point networks for large-scale point clouds. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11208, pp. 625–640. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01225-0_37
40. Riegler, G., Osman Ulusoy, A., Geiger, A.: OctNet: learning deep 3D representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3577–3586 (2017)
41. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
42. Su, H., et al.: Splatnet: sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
43. Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E.: Multi-view convolutional neural networks for 3D shape recognition. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 945–953 (2015)
44. Tatarchenko, M., Park, J., Koltun, V., Zhou, Q.Y.: Tangent convolutions for dense prediction in 3D. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3887–3896 (2018)
45. Te, G., Hu, W., Zheng, A., Guo, Z.: RGCNN: regularized graph CNN for point cloud segmentation. In: Proceedings of the 26th ACM International Conference on Multimedia, pp. 746–754 (2018)
46. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: KPConv: flexible and deformable convolution for point clouds. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 6411–6420 (2019)

47. Verma, N., Boyer, E., Verbeek, J.: Feastnet: feature-steered graph convolutions for 3D shape analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2598–2606 (2018)
48. Wang, C., Samari, B., Siddiqi, K.: Local spectral graph convolution for point set feature learning. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11208, pp. 56–71. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01225-0_4
49. Wang, L., Huang, Y., Hou, Y., Zhang, S., Shan, J.: Graph attention convolution for point cloud semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
50. Wang, S., Suo, S., Ma, W.C., Pokrovsky, A., Urtasun, R.: Deep parametric continuous convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018
51. Wang, X., He, J., Ma, L.: Exploiting local and global structure for point cloud semantic segmentation with contextual point representations. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 4573–4583 (2019)
52. Wu, B., Zhou, X., Zhao, S., Yue, X., Keutzer, K.: 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), pp. 4376–4382. IEEE (2019)
53. Wu, W., Qi, Z., Fuxin, L.: PointConv: deep convolutional networks on 3D point clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
54. Wu, Z., et al.: 3D shapenets: a deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1912–1920 (2015)
55. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Yu.: SpiderCNN: deep learning on point sets with parameterized convolutional filters. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11212, pp. 90–105. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01237-3_6
56. Yan, X., Zheng, C., Li, Z., Wang, S., Cui, S.: PointASNL: robust point clouds processing using nonlocal neural networks with adaptive sampling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2020
57. Yang, B., et al.: Learning object bounding boxes for 3D instance segmentation on point clouds. In: Advances in Neural Information Processing Systems (NeurIPS), pp. 6737–6746 (2019)
58. Yang, J., et al.: Modeling point clouds with self-attention and gumbel subset sampling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
59. Ye, X., Li, J., Huang, H., Du, L., Zhang, X.: 3D recurrent neural networks with context fusion for point cloud semantic segmentation. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 415–430. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_25
60. Yifan, W., Wu, S., Huang, H., Cohen-Or, D., Sorkine-Hornung, O.: Patch-based progressive 3D point set upsampling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019
61. Zhang, F., et al.: Instance segmentation of lidar point clouds. In: International Conference on Robotics and Automation (ICRA) (2020)

62. Zhang, Z., Hua, B.S., Yeung, S.K.: Shellnet: efficient point cloud convolutional neural networks using concentric shells statistics. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), October 2019
63. Zhao, H., Jiang, L., Fu, C.W., Jia, J.: Pointweb: enhancing local neighborhood features for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5565–5573 (2019)
64. Zhao, H., et al.: PSANet: point-wise spatial attention network for scene parsing. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11213, pp. 270–286. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01240-3_17
65. Zhao, Y., Birdal, T., Deng, H., Tombari, F.: 3D point capsule networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019