Pyramid Scene Parsing Network in 3D: improving semantic segmentation of point clouds with multi-scale contextual information

Hao Fang, Florent Lafarge

Pyramid Scene Parsing Network in 3D: improving semantic segmentation of point clouds with multi-scale contextual information

Hao Fang and Florent Lafarge

Inria, Université Côté d'Azur

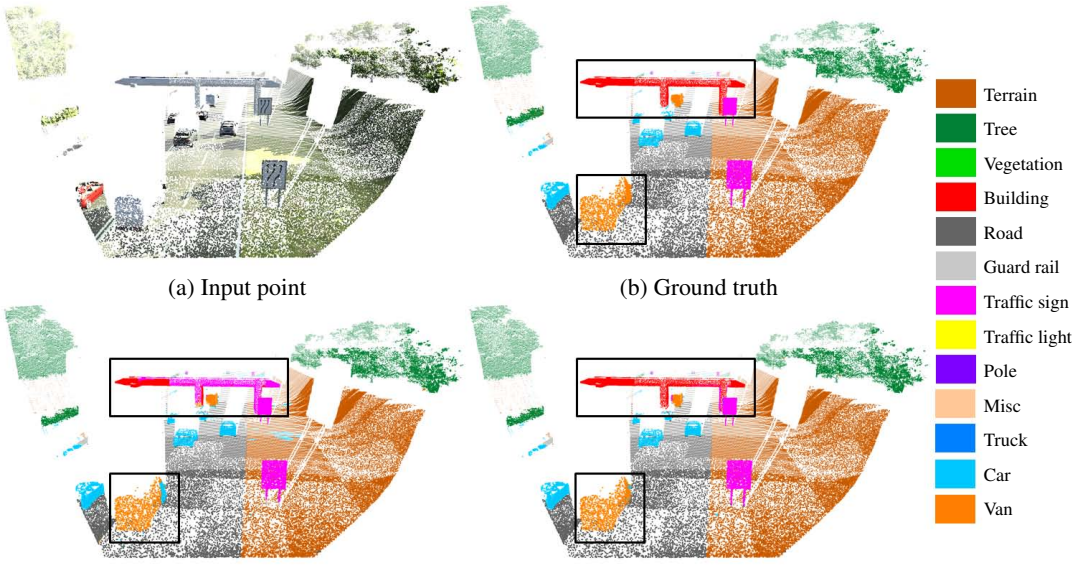
Abstract

Analyzing and extracting geometric features from 3D data is a fundamental step in 3D scene understanding. Recent works demonstrated that deep learning architectures can operate directly on raw point clouds, i.e. without the use of intermediate grid-like structures. These architectures are however not designed to encode contextual information in-between objects efficiently. Inspired by a global feature aggregation algorithm designed for images (Zhao et al., 2017), we propose a 3D pyramid module to enrich pointwise features with multi-scale contextual information. Our module can be easily coupled with 3D semantic segmantation methods operating on 3D point clouds. We evaluated our method on three large scale datasets with four baseline models. Experimental results show that the use of enriched features brings significant improvements to the semantic segmentation of indoor and outdoor scenes.

Keywords: Point Cloud, Semantic Segmentation, Deep Learning, Multi-scale Contextual Information

1. Introduction

The semantic segmentation of 3D point clouds is an important problem in 3D computer vision, in particular for autonomous driving, robotics and augmented reality (Tchapmi et al., 2017). Over the last decades, algorithms have mostly consisted in extracting low level features from point cloud via geometric prior knowledge (Nurunnabi et al., 2012; Lafarge and Mallet, 2012; Weinmann et al., 2013; Rouhani et al., 2017; Fang et al., 2018). The success of deep learning in image analysis (Long et al., 2015; Badrinarayanan et al., 2015; Chen et al., 2018) has drawn considerable attention in 3D scene understanding. Because point clouds



(a) Input point (b) Ground truth

(c) PointNet (d) PointNet + 3d-PSPNet

Figure 1: Semantic segmentation of a point cloud with and without our 3d-PSPNet module. Given an input point cloud (a), PointNet (Qi et al., 2017b) fails to predict correct labels for points describing large-scale objects (see rectangles in (c)). PointNet equipped with our 3d-PSPNet module gives better prediction results by enriching global contextual information (d).

are unstructured, it is not possible to use *convolutional neural network* (CNN) directly on such data for end-to-end training. An alternative approach is to first convert point clouds into an intermediate grid-like representation before exploiting CNNs. Such representations can take the form of multi-view images (Su et al., 2015; Kalogerakis et al., 2017; Boulch et al., 2018) or voxel grids (Maturana and Scherer, 2015; Wu et al., 2015; Qi et al., 2016; Tchapmi et al., 2017; Hackel et al., 2017). However, these underlying representations typically lead to a loss of 3D information and are often not memory-efficient. The PointNet architecture proposed by Qi et al. (2017b) uses a composition of basic operators, i.e. *multi-layer perceptron* (MLP) and *max pooling* to extract features directly from point clouds. While simple, this architecture learns order-invariant pointwise features and exhibits good performance on various tasks. This work has inspired several approaches for enriching pointwise features by aggregating information in local regions (Qi et al., 2017c; Huang et al., 2018; Wang et al., 2018b). However, the receptive field of these methods is not effective when objects of different scales

are close to each other. Zhao et al. (2017) demonstrated that a prior knowledge of global regional context is crucial for improving prediction accuracy.

In this paper, we address the problem of increasing the receptive field of points by inferring regional global contextual information. Inspired by Zhao et al. (2017), we design a *3D pyramid scene parsing network* (3d-PSPNet) to enrich local pointwise feature with multi-scale global contextual information. We evaluated the impact of our 3d-PSPNet module on three datasets and four baseline networks. Experimental results show that the enriched features provide more accurate predictions than by using the baseline models only (see Figure 1). The goal of our work is not to achieve state-of-the-art performances on the datasets, but to propose a generic module that can be concatenated with any 3D neural network to infer richer pointwise features. Meanwhile, we provide all the important training and testing details throughout our experiments and make our code public.

2. Related Work

The success of CNNs with images (Krizhevsky et al., 2012; Long et al., 2015; Badrinarayanan et al., 2015; Chen et al., 2018) encouraged researchers to adapt these tools to 3D data. We review recent deep learning approaches for different 3D data representations, including *multi-view images*, *voxels* and *point clouds*.

2.1. Multi-view images

A straightforward idea is to represent 3D data with a set of multi-view images that can be directly processed by CNNs. Su et al. (2015) proposed such an architecture without inserting global consistency among all images. More recently, Kalogerakis et al. (2017) employ image-based *Fully Convolutional Network (FCN)* for part-based mesh segmentation. Similarly, Boulch et al. (2018) propose a FCN for 2d image semantic segmentation where the output labels are then back projected to the original point cloud. This strategy benefits from the efficiency of CNNs for feature learning, but typically suffers from a loss of information when multiple 2D representations are projected into the 3D space.

2.2. Voxels

Another popular representation consists in converting 3D data into a voxelized occupancy grid (Maturana and Scherer, 2015). Wu et al. (2015) proposed a 3D-CNN based framework for object category recognition and shape completion. Qi et al. (2016) exploited two network architectures of volumetric CNNs to improve the performance of both voxel based and multi-view based approaches. Hackel

et al. (2017) and Roynard et al. (2018) designed a multi-scale 3D-CNN network for large scale urban scene segmentation. Similarly, Tchapmi et al. (2017) involved a 3D-FCN model with a post-processing of voxel-based predictions inspired from Zheng et al. (2015). Dai and Nießner (2018) designed a joint 2D-3D network to first analyze multi-view RGB images and then map the features back to a voxel grid. This joint 2D-3D method incorporates both RGB features and geometric features, and yields more accurate prediction result. Zeng et al. (2017) and Gojcic et al. (2018) extracted local 3D volumetric patches and learned local geometric descriptors for characterizing correspondences between 3D point clouds. These different frameworks produce good results on large scale datasets. However, they suffer from high memory consumption. In practice, the size of the grid is typically limited to $100 \times 100 \times 100$ voxels. To tackle this problem, several works focused on reducing the computational burden caused by sparsity of grid occupancy by employing spatially-adaptive data structures (Riegler et al., 2017; Klokov and Lempitsky, 2017).

2.3. Point clouds

The main challenge for using CNNs directly on point clouds is to design an order-invariant and differentiable feature extraction operator that can be trained end-to-end. Qi et al. (2017b) proposed PointNet, a powerful neural network composed of a stack of basic operators that can handle unstructured point cloud directly. The main idea is to process each point independently with a sequence of *multi layer perceptrons* (MLP) with shared weights for all points. The learned pointwise features are typically aggregated into a global feature for classification tasks. Inspired by this architecture, many works focused on learning richer pointwise features by incorporating local dependencies in the local neighborhood of each point (Qi et al., 2017c; Huang et al., 2018; Wang et al., 2018b; Su et al., 2018; Jiang et al., 2018). Atzmon et al. (2018) first introduced an extension operator to map point-based functions to volumetric functions and then restricted them back to the point cloud. Xu et al. (2018) proposed new parameterized convolutional filters to model the intersection between irregular point sets. Rethage et al. (2018) proposed a hybrid method by first employing PointNet as low-level feature descriptor and then converting points into internal representations. A fully-convolutional network is finally used to learn multi-scale features. These methods achieve good performances on various 3D classification and semantic segmentation datasets. In contrast, Landrieu and Simonovsky (2018) first partitioned the points into different clusters by constructing a supergraph of points. The semantic label of each cluster is estimated by PointNet and the final result is obtained

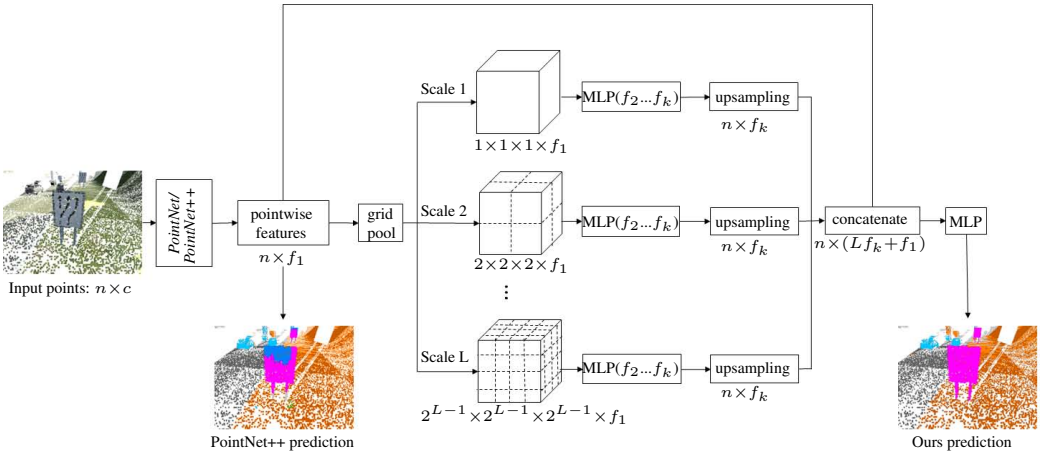


Figure 2: Diagram of our 3d-PSPNet module. We depart from PointNet (Qi et al., 2017b) or PointNet++ (Qi et al., 2017c) to first capture pointwise local features. A pyramid structure model is then used to exploit multi-scale global contextual features in each sub-region. Finally, the local pointwise features and learned multi-scale contextual features are concatenated together. The enriched features give more accurate results than by using baseline model only. In particular, PointNet++ fails at the top part of the *traffic-sign*. The mislabeled points are corrected with our 3d-PSPNet module.

after graph regularization (Simonovsky and Komodakis, 2017). Besides, PointNet also serves as a general pointwise feature extraction tool for other tasks, including 3D object detection (Qi et al., 2017a), point cloud upsampling (Yu et al., 2018), instance segmentation (Wang et al., 2018a) and 3D reconstruction (Groueix et al., 2018). Other methods (Deng et al., 2018b,a) represented patches of points by low-level features and then enriched local descriptors used for 3D matching. Zhao et al. (2018) designed a 3D Capsule-encoder for point clouds used for 3D reconstruction, 3D matching and segmentation. The design of effective receptive fields for point clouds has not been widely studied in the literature. None of these methods truly extract effective global contextual information to enrich pointwise features.

3. Overview

We propose a generic 3d-PSPNet module that can be concatenated after any pointwise feature based approach. Our goal is to enlarge the receptive field of points by inserting multi-scale contextual information in sub-regions. The paper

is organized as follows. Section 4 presents the architecture of 3d-PSPNet. Experimental results on three large scale datasets with four baseline models are then presented and discussed in Section 5.

4. Proposed method

Let us consider an input point cloud $P = \{p_1, p_2, \ldots, p_n\}$, where each point $p_i \in \mathbb{R}^c$ is defined by its spatial coordinates and a set of extra information such as color and normal. c is the number of input features. Deep neural networks such as pointNet (Qi et al., 2017b) and PointNet++ (Qi et al., 2017c) allow us to compute the pointwise feature set $F = \{f(p_1), f(p_2), \ldots, f(p_n)\}$, where $f(p_i) \in \mathbb{R}^{f_1}$ is an f_1-dimensional feature vector. Our objective is to discover global contextual clues for each point and return the enriched pointwise feature set $\hat{F} = \{\hat{f}(p_1), \hat{f}(p_2), \ldots, \hat{f}(p_n)\}$. Inspired by the work of Zhao et al. (2017) in image analysis, we exploit global contextual features at multiple scales. Figure 2 illustrates the diagram of the network.

4.1. Network architecture.

Our 3d-PSPNet exhibits a pyramid structure. At each pyramid scale l, we extract contextual clues by 3 basic operations.

- **Grid pooling.** Given input points $p_1, p_2, \ldots, p_n$ and pointwise features $f(p_1), f(p_2), \ldots, f(p_n)$ returned by PointNet or PointNet++, this step projects each point to a local sub-region. More specifically, we first split the whole scene into $2^{l-1} \times 2^{l-1} \times 2^{l-1}$ voxelized cells. Points are then grouped into the corresponding grid according to their spatial position in the scene. This basic operation enables each point to exploit contextual information in its sub-region independently. Note that we save the voxel index where each point is projected to as $G = \{g(p_1), g(p_2), \ldots, g(p_n)\}$ for further processing. Finally, a basic max pooling layer is used on all the points of the grid (i, j, k). We obtain a f_1-dimensional global feature vector f^l_{ijk}. Our grid pooling layer outputs a $2^{l-1} \times 2^{l-1} \times 2^{l-1} \times f_1$ tensor at scale l.

  Another choice for grouping the points into different 3D grids is the use of *sampling layer* proposed by Qi et al. (2017c). This fine-to-coarse approach selects a fixed number of most distant points using iterative *farthest point sampling* (FPS). Then, the other points are clustered into the group of selected point according to query ball or k-nearest neighbors. Compared to a voxel grid grouping, this choice requires longer processing time to select

the distant points and preserves unstructured grid shape. However, it solves the sparsity problem of point clouds, ensuring that each sub-regional grid have some projected points inside. We call this architecture the *adaptive grid* method. We discuss the comparison between these two architectures in Section 5.6.

- **Sub-regional feature aggregation.** Our grid pooling operator projects all points and corresponding features onto different sub-regions. Then, we enhance each sub-regional global feature f^l_{ijk} via a sequence of MLP with output channels $(f_2, \ldots, f_d)$. Enriched global feature at grid (i, j, k) is now a f_k-dimensional feature vector $\hat{f}^l_{ijk}$. Note that all the cells at each pyramid scale l share the same MLP weights, which preserves the order-invariant property of our network. This step outputs a $2^{l-1}\times 2^{l-1}\times 2^{l-1}\times f_d$ tensor at scale l.

- **Grid upsampling.** The previous step provides an enhanced global contextual vector $\hat{f}^l_{ijk}$ for each grid, which serves as a representable clue for all the points inside each sub-region. To output an enriched feature for all the points, we use the point-to-cell assignments, denoted as G, to upsample all the points in each grid and assign them a global contextual feature of its corresponding grid so that $\hat{f}^l(p_i) = \hat{f}^l_{g(p_i)}$. Finally, this step output the enhanced sub-regional contextual pointwise feature set $\hat{F}^l = \{\hat{f}^l(p_1), \ldots, \hat{f}^l(p_n)\}$.

We then group enriched pointwise features of all pyramid scales together as the multi-scale contextual feature set $\hat{F}^C = \hat{F}^1 \oplus \hat{F}^2 \oplus \cdots \oplus \hat{F}^L$, where $\oplus$ is the concatenation operator. We argue that the enriched pointwise feature set $\hat{F}$ extracts multi-scale contextual information by aggregating features learned at different sub-regions with varied sizes. Finally, we assemble contextual and local features together to not lose the pre-learned local information. The final enriched pointwise feature set is then $\hat{F} = \hat{F}^C \oplus F$. Note that our 3d-PSPNet can enrich pointwise features that are invariant to the order of input points. Because the main application of our 3d-PSPNet is the semantic segmentation of indoor and urban scenes, we use a *cross-entropy* loss function in the framework.

5. Experiments

We evaluated our approach on three datasets: S3DIS (Armeni et al., 2016), ScanNet (Dai et al., 2017) and vKITTI (Francis et al., 2017). The two first datasets

propose large collections of indoor scenes whereas the last one focuses on urban scenes from street-side acquisition. For each dataset, we compared the results with and without using our 3d-PSPNet module for four state-of-the-art baselines: PointNet (Qi et al., 2017b) PointNet++ (Qi et al., 2017c), DGCNN (Wang et al., 2018b) and PointSIFT (Jiang et al., 2018). To fairly measure the impact of our module, we used the same parameters and hyperparameters for each baseline and its enriched version. All the experiments were performed on a NVIDIA GeForce GTX 1080 Ti GPU.

5.1. Implementation details

Unified diagram. The main hyperparameters involved in our network are the number of pyramid scales L, the dimension of feature aggregation fully connected layers MLP $(f_2, \ldots, f_k)$, and the number of grids at each scale. Remind that our goal is not to achieve state-of-the-art performance on each dataset, but to increase the accuracy of existing 3D neural networks. Therefore, we keep all the parameters and hyperparameters fixed and only compare the results with and without using our module. We use one unified diagram in all the experiments. First, we observe that the number of pyramid scales L is a trade-off between prediction accuracy and computational efficiency. Choosing a high value for L gives more accurate results but strongly increases the training time. In our experiments, we use a 4-level pyramid structure. We detail the choice of L in Section 5.6. Second, empirical results demonstrate that increasing the number of MLP in the sub-regional feature aggregating layer does not necessarily improve accuracy. We choose a 256-channel MLP to aggregate the global contextual information in each sub-region. This choice avoids considering numerous parameters to be learned and also prevents the network from overfitting. Batch normalization and Relu activations are involved after each MLP. Finally, we set the number of grids in each dimension (x, y, z) at each scale l to 2^{l-1} in order to preserve a multi-scale pyramid structure.

Training strategy. We followed the data-preparing process proposed by Qi et al. (2017b) for all the datasets. Scenes were first divided into a set of blocks of similar size. Then, a fixed number of points were sampled from each block to make the training more efficient. Each block served as a mini-batch for the end-to-end training. Finally, each trained model was used to test blocks for final semantic segmentation evaluation. For a fair comparison, we followed the same training details than the four baselines (Qi et al., 2017b,c; Wang et al., 2018b; Jiang et al., 2018). We used Adam optimizer with initial learning rate to 0.001.

The learning rate was divided by 2 every 300000 mini batches for S3DIS and ScanNet, and every 200000 mini batches for vKITTI.

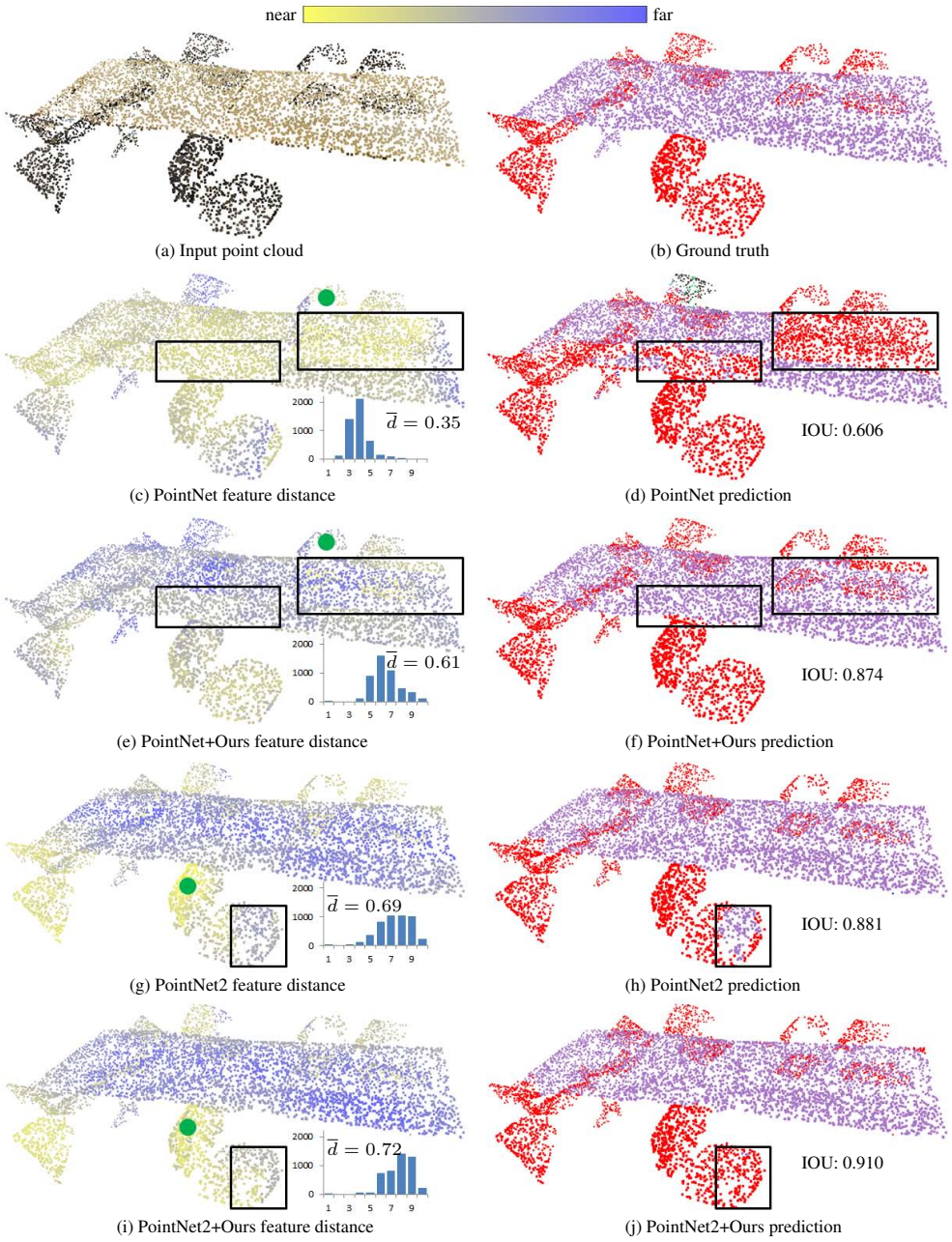
Evaluation metrics. We evaluated the prediction results with both qualitative and quantitative comparisons. For the quantitative evaluation, we used three standard semantic segmentation evaluation metrics: *Overall* Accuracy (OA), *mean Intersection Over Union* (mIOU) and *mean Accuracy Over Classes* (mAcc). The formulation of these metrics can be found in (Tchapmi et al., 2017).

5.2. Feature analysis

With our 3d-PSPNet, the baseline models PointNet and PointNet++ are able to learn enriched pointwise feature by incorporating both local and multi-scale global information. Figure 3 analyzes the quality of learned features on these two baseline models with and without our module (for clarity issue, we replace PointNet++ by PointNet2 in the following). We output the features learned at the last layer, and visualize extracted features as the Euclidean distance from every point to a standard point with ground truth label *chair* in feature space. The color of each point varies from yellow to blue, representing a near-to-far feature distance from current point to the selected standard point. Besides, we compute the distribution of feature distance from points with label *table* to standard point with label *chair* (see blue histograms in Figure 3 - left).

As a result, Figure 3c shows that *table* points have mostly a relatively close distance to *chair* point with a mean distance to 0.35. This observation means that features learned by PointNet baseline fail to clearly discriminate *table* and *chair*. By using our 3d-PSPNet after the PointNet baseline, the mean distance increases to 0.61. In addition, the feature distance distribution shifts from lower bins to higher bins (see Figure 3- e and c). Points with label *chair* are close to standard point (green spot in Figure 3) while points with label *table* are far away from standard point in feature space. Consequently, the *Intersection Over Union* (IOU) score increases from 0.606 to 0.874 by using our 3d-PSPNet (see prediction differences between Figure 3d and 3f).

Because the hierarchical architecture of PointNet2 baseline produces richer local features than PointNet, the mean feature distance from *table* points to standard point reaches a higher value, i.e. 0.69 (see Figure 3g). It also achieves more accurate segmentation results with an IOU to 0.881. Yet, a small part of points with label *chair* is still relatively far away from standard point in feature space (see rectangle in Figure 3g). In addition, the network mislabeled them to *table* (see Figure 3h). When using our 3d-PSPNet module with PointNet2, these mislabeled points have a lower feature distance to standard point with a mean feature



(a) Input point cloud (b) Ground truth

(c) PointNet feature distance (d) PointNet prediction

(e) PointNet+Ours feature distance (f) PointNet+Ours prediction

(g) PointNet2 feature distance (h) PointNet2 prediction

(i) PointNet2+Ours feature distance (j) PointNet2+Ours prediction

Figure 3: Feature analysis on an indoor scene composed of two types of objects: *chair* (red) and *table* (purple). Given an input scene (a), we extract features of the output layer learned by each model and visualize the feature distance from each point to a standard point (green spot) with ground truth label *chair*. Models equipped with our 3d-PSPNet not only reduce the feature distance from points with label *table* to standard point (see shift of blue histograms from lower bins to higher bins in (e) and (i) compared with (c) and (g)), but also produce better prediction results than by using the baseline only (see chair in (d, f) and (h, j)).

distance to 0.72 (see Figure 3i). Figure 3j shows the prediction result for these mislabeled points. The IOU score in this case is 0.91.

5.3. Semantic Segmentation on the S3DIS Dataset

We evaluated our 3d-PSPNet module on the S3DIS dataset. The whole dataset contains 6 areas with 271 rooms and 13 classes. Each point has an annotation label from 13 classes. As proposed in (Qi et al., 2017b), each room is split into blocks with size $1m \times 1m$ in x and y directions with a $0.5m$ stride in the training procedure. To make batch training available and accelerate the training process, we sample 4096 points in each block. Every sampling point contains 9 dimensional channels: $[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$, representing position, color and normalized position of each point in the current room. In the testing process, all rooms are split into non-overlapping blocks with size $1m \times 1m$. Following the 1-fold experimental protocol described in (Armeni et al., 2016), we tested on Area 5 and trained on the other areas.

We evaluated the benefit of our 3d-PSPNet module with PointNet, PointNet2, DGCNN and PointSIFT baseline models through control experiments. For the PointNet baseline, we set batch size to 24 and assigned a 9-dimensional input feature to each point $[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$. For the PointNet2 baseline, the batch size was also 24 but we assigned a 3-dimensional channel $[x, y, z]$ as input feature to each point. This setting is designed to exhibit the generalization of our 3d-PSPNet in case of insufficient input features. Due to limited computational resources, we restricted the batch size of DGCNN and PointSIFT baselines to 12. Although this choice may not achieve the best performance for these baseline models, we fixed this setting in control experiments for a fair comparison. We assigned $[x, y, z]$ as input features to each point for these two models. No data augmentation technique was used in these experiments. Note also that all the hyperparameters were fixed in these control experiments.

PointNet and PointNet2 baseline models with and without our 3d-PSPNet module were trained from scratch for 50 epochs. We plot training and testing errors on Area 5 along all epoch for these 4 models in Figure 5. The gap between the baseline curve and our curve means that our 3d-PSPNet module improves both training and testing accuracy along the whole training procedure. Although there is an oscillation along the testing accuracy curves, our model finally converges to an optimal with a lower testing error, and thus increases the generalization of the two baseline models. Figure 4 shows visual results on two indoor scenes. PointNet and PointNet2 baseline provide accurate prediction for a majority of points. Our 3d-PSPNet succeeds in correcting prediction results for mislabeled points,

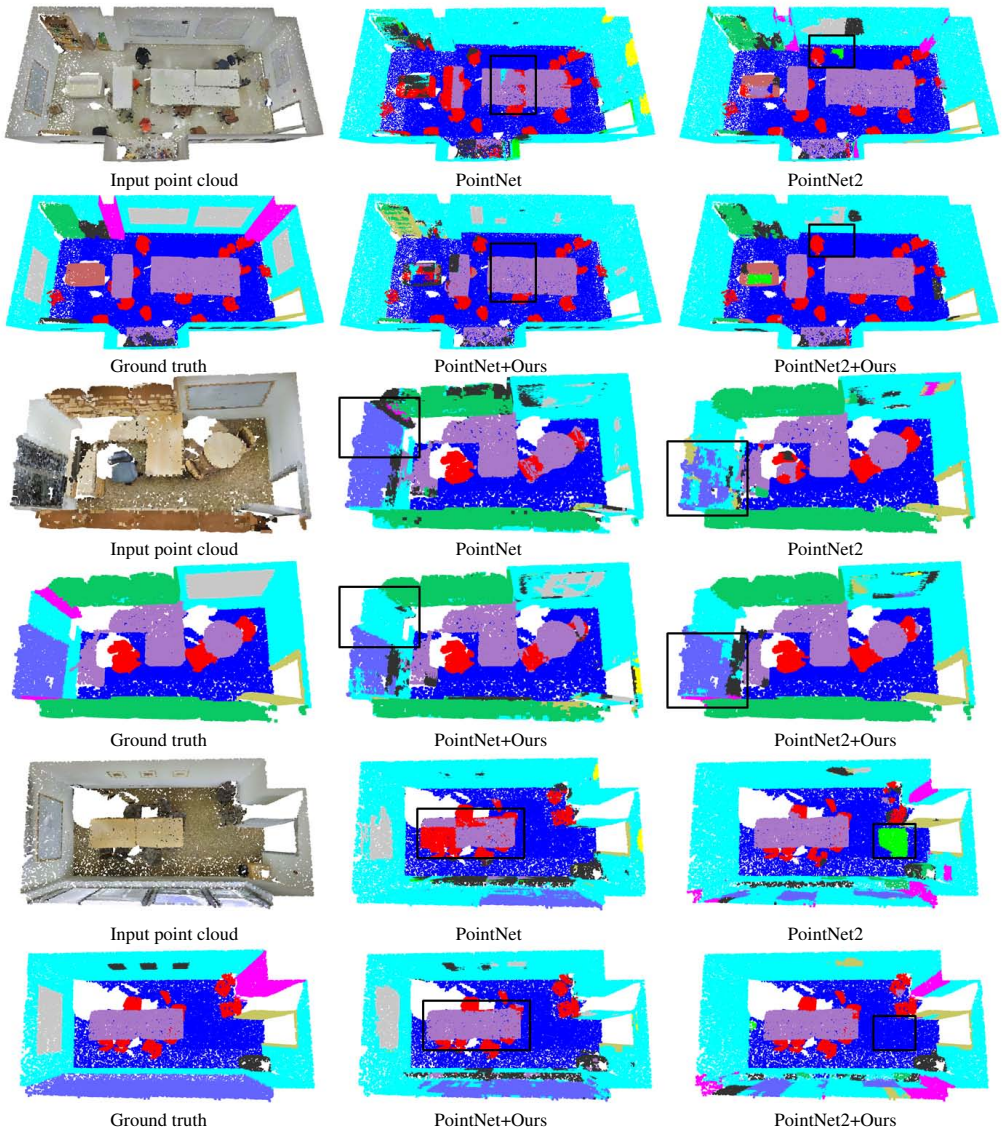


Figure 4: Qualitative results on S3DIS dataset. The use of our 3d-PSPNet module gives better prediction results than by using the baseline model only (see black boxes).

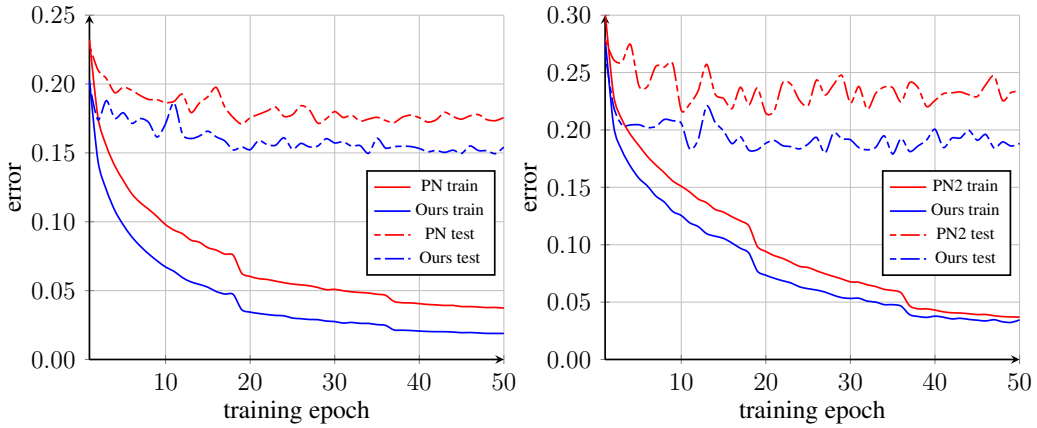


Figure 5: Training error and testing error on S3DIS dataset Area 5 of PointNet (abbreviated as PN, left image) and PointNet2 (abbreviated as PN2, right image) baseline with and without our 3d-PSPNet learned from scratch. Note that our 3d-PSPNet improves OA by 2.27% and 3.78% for PointNet and PointNet2 respectively.

i.e. *window* and *table*. In addition, the quantitative segmentation measurement on S3DIS dataset Area 5 is given in Table 1. Our 3d-PSPNet module improves mIOU by 4.52% (respectively 4.96%) and mAcc by 5.57% (resp. 4.82%) for PointNet (resp. PointNet2) baseline. IOU of 10 and 11 out of 13 classes are improved for PointNet and PointNet2 baseline respectively. According to qualitative and quantitative comparisons, our 3d-PSPNet reinforces the generalization of baseline models.

Considering the large scale training parameters of DGCNN and PointSIFT baseline models, we exploited a *transfer learning* strategy on the training phase to improve the efficiency of the training procedure. In practice, we first trained the DGCNN and PointSIFT baselines for 25 epochs. To perform a fair comparison, we used the pre-trained models to initialize the weights of parameters in the network and continued the experiments along two different paths. First, we concatenated our 3d-PSPNet module after the pre-trained models and fine tuned the whole network for 25 epochs. Second, we continued training the pre-trained models without our 3d-PSPNet module for 25 epochs. Quantitative results are shown in Table 1. Our 3d-PSPNet module improves mIOU by 2.68% (respectively 4.96%) and mAcc by 1.97% (resp. 4.34%) for the DGCNN (resp. PointSIFT) baseline. Also, prediction results of 11 and 12 out of 13 classes were improved.

Method	mIOU	mAcc	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet	41.02	48.51	89.41	**98.35**	69.14	0.04	**5.54**	45.20	**11.56**	58.69	53.21	2.90	42.63	23.09	33.47
PointNet+Ours	**45.54**	**54.08**	**92.05**	97.59	**70.60**	**0.43**	5.04	**49.83**	7.73	**64.82**	**68.71**	**11.36**	**47.28**	**38.35**	**38.27**
PointNet2	43.11	53.39	71.80	74.75	69.35	0.00	11.70	22.15	42.92	59.93	75.71	22.63	**51.74**	17.51	40.23
PointNet2+Ours	**48.07**	**58.21**	**79.99**	**84.15**	**73.32**	0.00	**20.21**	**32.69**	**50.25**	**62.02**	**78.25**	**31.02**	51.24	**21.04**	**40.68**
DGCNN	46.29	55.87	90.88	**97.58**	69.27	0.00	19.47	**30.93**	47.99	67.29	68.80	23.61	40.14	8.40	37.39
DGCNN+Ours	**48.97**	**57.84**	**91.51**	97.55	**73.49**	**0.12**	**21.24**	20.80	**60.44**	**69.21**	**72.98**	**30.12**	**44.26**	**14.35**	**40.58**
PointSIFT	45.66	56.65	68.72	73.33	73.44	0.00	6.77	27.57	46.79	52.08	82.25	36.86	50.34	27.39	48.07
PointSIFT+Ours	**50.62**	**60.99**	**78.74**	**82.64**	**74.08**	0.00	**19.67**	**35.06**	**53.15**	**58.92**	**83.65**	**42.35**	**51.55**	**32.09**	**46.22**

Table 1: Quantitative results on S3DIS dataset Area 5, including mIOU, mAcc and IOU for 13 classes.

5.4. Semantic Segmentation on the ScanNet Dataset

We next evaluated our 3d-PSPNet on the ScanNet dataset. This large scale indoor dataset contains 1201 training rooms and 312 testing rooms with 21 classes including an *unannotated* class. We split each room into blocks of size $1m \times 1m$ in x and y directions with a $1m$ stride (instead of $0.5m$ in the training procedure). Each block contains 4096 sampling points. This choice avoids consuming too much training time on large scenes. Remind that every sample point contains 9-dimensional channel: $[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$. For testing, we applied the trained model on testing blocks with size of $1m \times 1m$.

We followed the experiment settings used in Section 5.3 by concatenating our 3d-PSPNet module after PointNet, PointNet2 and DGCNN baselines. However, considering the big size of training data, we followed the *transfer learning* strategy proposed in Section 5.3. In practice, we first trained the baseline models for 20 epochs before stopping. To perform a fair comparison, we used the pre-trained models to initialize the weights of parameters in the network and continued the experiments along two different paths. First, we concatenated our 3d-PSPNet module after the pre-trained models and fine tuned the whole network for 20 epochs. Second, we continued training the pre-trained models without our 3d-PSPNet module for 20 epochs.

Figure 7 compares training and testing errors along the last 20 training epochs of PointNet and PointNet2. By using PointNet and PointNet2 models only, the training curves progressively converge while testing curves increase along the training process. This phenomenon results from the overfitting of baseline models to the training set. However, the use of our 3d-PSPNet module after these baseline models improves the testing accuracy from first epoch of fine-tuning and and makes the process converge in only a few epochs. This observation shows that our 3d-PSPNet raises the generalization of baseline models. For PointNet and PointNet2, our module improves OA by 1.26% and 2.19% respectively (by including the *unannotated* class). Figure 6 shows some qualitative results. Note

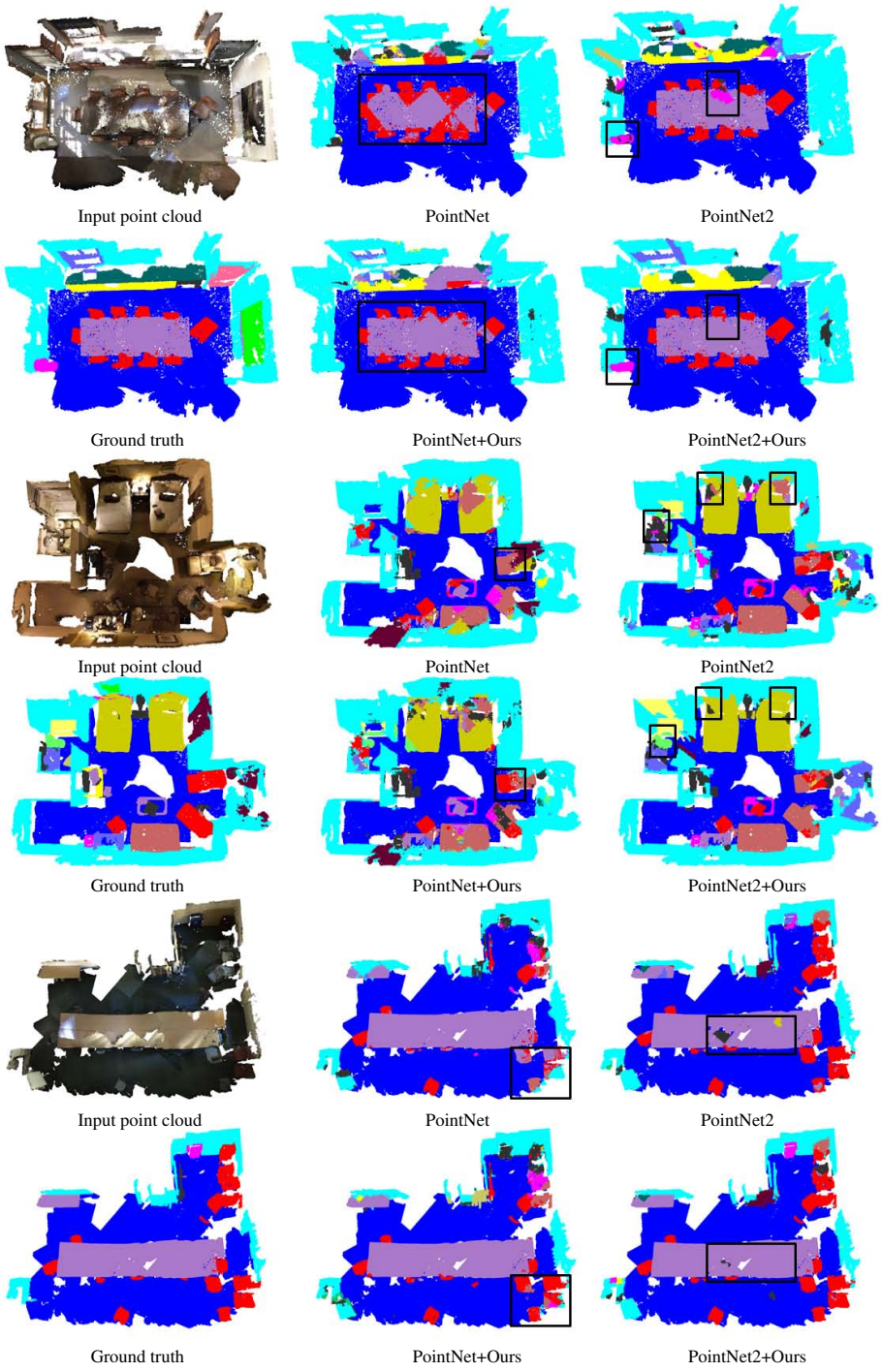


Figure 6: Qualitative results on ScanNet dataset. Our 3d-PSPNet improves the prediction results returned by baseline models via transfer learning strategy (see changes in black boxes).

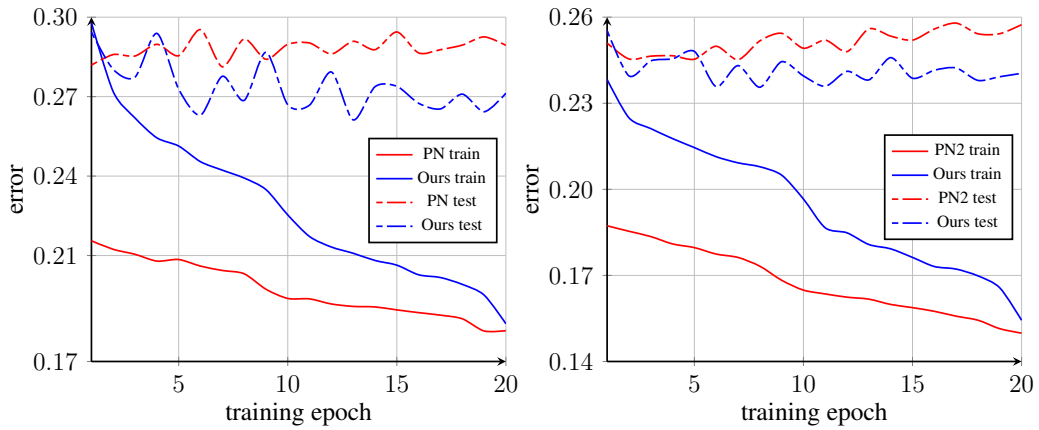


Figure 7: Training and testing error of PointNet (abbreviated as PN, left image) and PointNet2 (abbreviated as PN2, right image) baselines with and without our 3d-PSPNet module on the ScanNet dataset. The network with our 3d-PSPNet improves OA by 1.26% and 2.19% for PointNet and PointNet2 respectively.

that, to perform a fair visual comparison, we show the results given by baseline models trained after the first 20 epochs. Our 3d-PSPNet module corrects some mislabeled points returned by baseline models and makes prediction results more consistent. This modification benefits from fine-tuning training strategy where mislabeled points incorporate pyramid contextual information from its local regions. Table 2 presents the quantitative results on the testing rooms. Our 3d-PSPNet increases mIOU by 3.17% (respectively 2.04% and 3.51%) and mAcc by 5.02% (resp. 5.80% and 5.31%) for PointNet (resp. PointNet2 and DGCNN). IOU of 18, 16 and 19 out of 20 classes are improved for PointNet, PointNet2 and DGCNN baselines respectively.

We tested our 3d-PSPNet on different experiment settings. We followed the data preparing method proposed by Dai et al. (2017) for PointSIFT baseline. In the training phase, each scene was divided into $1.5m \times 1.5m \times 3m$ voxelized cubes. We kept the cubes where more than 2% of the voxels were occupied by points. In the testing phase, each scene was split into smaller cubes and we returned the labeling results of each point in the cubes. Only the $[x, y, z]$ information were considered for each point. The baseline model was trained for 200 epochs. Then we concatenated our 3d-PSPNet after the pre-trained model and fine tuned the whole network for 200 epochs. In the control experiment, we continued training the pre-trained baseline model for 200 more epochs. Table 3 shows that our 3d-PSPNet achieves 3.03% improvement for mIOU and 2.98% improvement for mAcc.

Method	mIOU	mAcc	wall	floor	chair	table	desk	bed	book-shelf	sofa	sink
PointNet	23.64	33.25	67.48	87.70	41.31	40.48	14.83	32.85	19.73	28.12	14.86
PointNet + ours	**26.81**	**38.27**	**69.33**	**89.38**	**44.24**	**44.25**	**16.34**	**35.85**	**27.47**	**29.88**	**19.64**
PointNet2	30.98	42.40	**71.55**	87.59	57.17	45.94	14.66	40.83	31.92	42.52	17.61
PointNet2 + ours	**33.02**	**48.20**	71.42	**88.75**	**57.67**	**48.09**	**18.87**	**41.91**	**36.77**	**46.53**	**21.68**
DGCNN	27.51	39.72	68.86	89.17	47.26	44.66	18.41	38.07	28.50	29.36	14.46
DGCNN + ours	**31.02**	**45.03**	**72.40**	**89.72**	**51.00**	**46.80**	**20.63**	**39.18**	**30.43**	**37.01**	**17.35**

Method	bathtub	toilet	curtain	counter	door	window	shower curtain	refrid-gerator	picture	cabinet	other furniture
PointNet	24.19	22.27	8.49	11.64	12.05	9.02	4.31	**7.98**	0.87	17.02	**7.53**
PointNet + ours	**36.63**	**28.54**	**10.55**	**13.45**	**13.01**	**12.39**	**9.53**	7.90	**3.58**	**17.89**	6.44
PointNet2	**49.15**	**33.06**	25.72	14.54	13.37	8.07	18.05	15.85	**0.76**	18.35	12.85
PointNet2 + ours	48.27	32.51	**26.87**	**17.81**	**14.53**	**9.75**	**18.54**	**26.01**	0.47	**20.81**	**13.09**
DGCNN	**37.80**	23.99	8.60	14.44	14.75	12.14	11.86	17.44	2.65	17.30	10.48
DGCNN + ours	37.57	**36.78**	**15.07**	**15.50**	**17.61**	**15.35**	**17.16**	**21.12**	**6.51**	**22.16**	**11.16**

Table 2: Quantitative results on the ScanNet dataset for PointNet, PointNet2 and DGCNN, including mIOU, mAcc and IOU for 20 classes. Note that the *unannotated* class is not considered in this evaluation.

Method	mIOU	mAcc	wall	floor	chair	table	desk	bed	book-shelf	sofa	sink
PointSIFT	40.01	57.33	76.02	89.02	71.99	39.26	25.06	50.22	24.91	58.43	37.07
PointSIFT + ours	**43.04**	**60.31**	**76.04**	**89.26**	**72.67**	**46.88**	**26.15**	**50.49**	**30.33**	**59.82**	**43.32**

Method	bathtub	toilet	curtain	counter	door	window	shower curtain	refrid-gerator	picture	cabinet	other furniture
PointSIFT	61.93	67.12	18.85	**21.19**	**21.05**	**19.23**	30.63	**48.62**	0.00	28.18	11.53
PointSIFT + ours	**69.30**	**70.20**	**35.03**	19.76	18.68	16.95	**35.02**	45.38	**2.53**	**34.65**	**18.36**

Table 3: Quantitative results on the ScanNet dataset for PointSIFT, including mIOU, mAcc and IOU for 20 classes. Note that the *unannotated* class is not considered in this evaluation.

5.5. Semantic Segmentation on the vKITTI Dataset

We finally evaluated the impact of our module on the vKITTI dataset which is composed of point clouds obtained by Velodyne LiDAR scanners. The whole dataset contains 6 non-overlapping urban scenes with 13 classes. Outdoor urban scenes usually contain larger objects than indoor scenes with cars, buildings etc. We thus split each scene into non-overlapping blocks of size $5m \times 5m$ in x and y directions to make sure that the large scale objects are split into a low number of blocks. We sampled 2048 points in each block as the mini training batch. Every sample point contains 9-dimensional channel: $[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$. During testing, we applied the trained model on all the testing blocks. We followed the 6-fold cross validation protocol described in (Francis et al., 2017).

Similarly to previous experiments, we performed four pairs of comparisons with PointNet, PointNet2, DGCNN and PointSIFT baseline models. Because

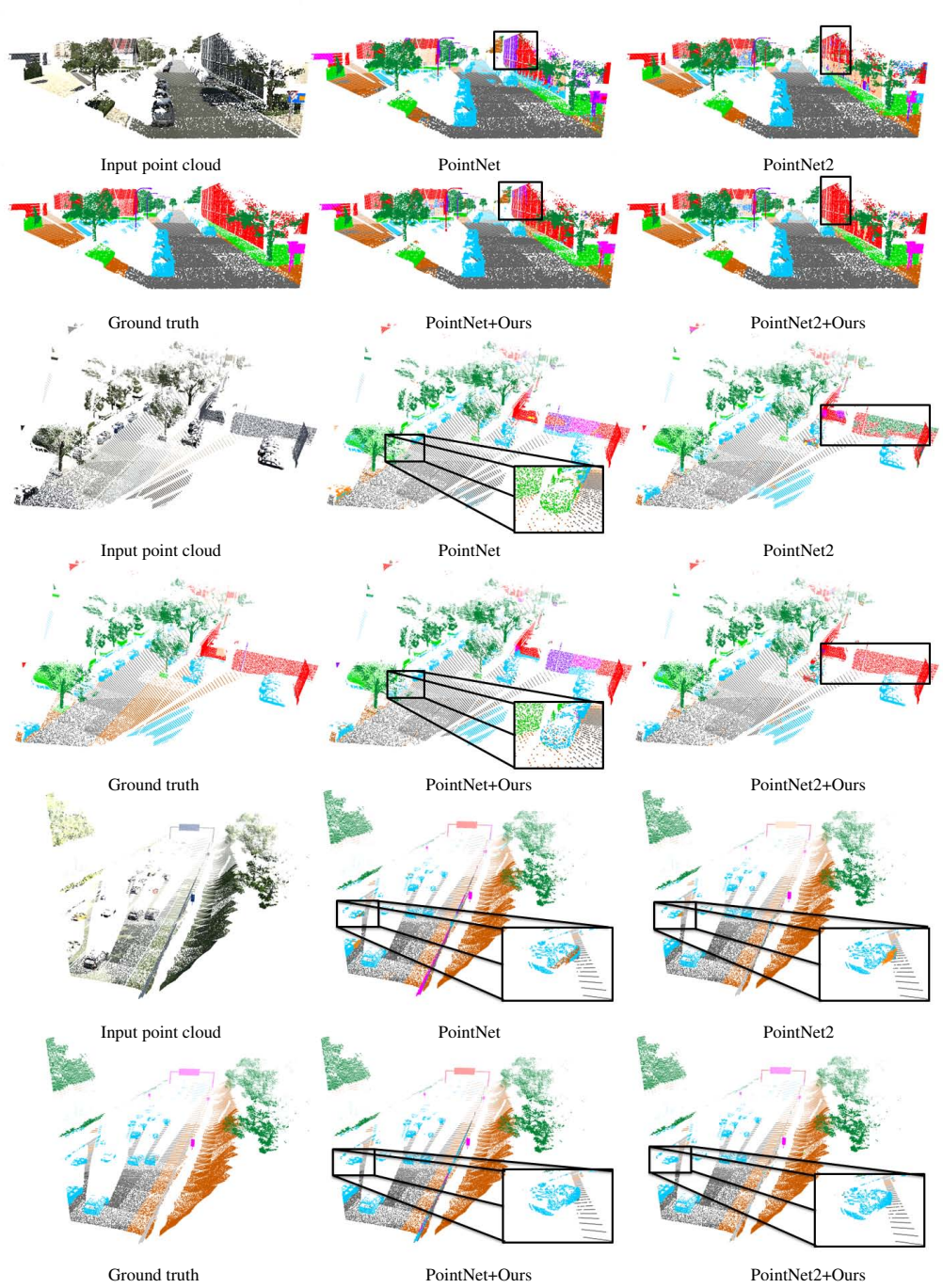


Figure 8: Qualitative results on vKITTI dataset. Our 3d-PSPNet module corrects some mislabeled points in the baseline models.

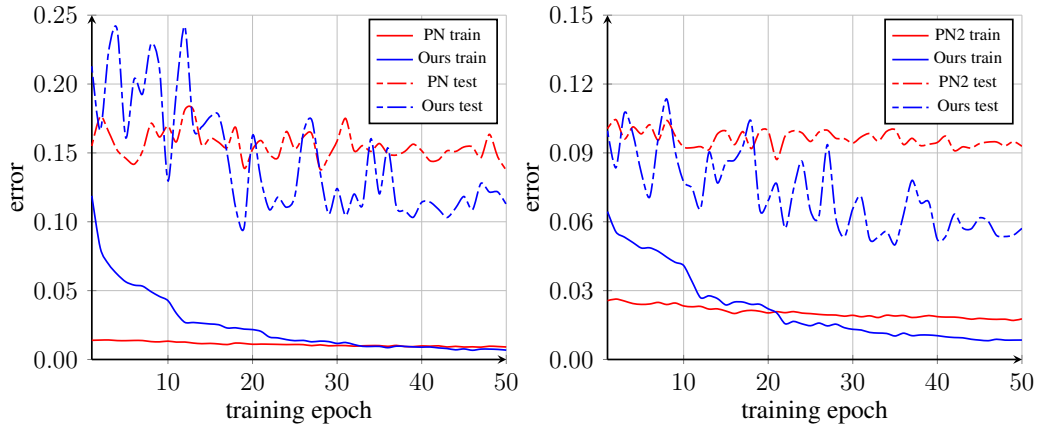


Figure 9: Training and testing errors of PointNet (abbreviated as PN, left image) and PointNet2 (abbreviated as PN2, right image) baselines with and without our 3d-PSPNet on Scene 6 of vKITTI dataset. With our 3d-PSPNet module, the network improves OA by 2.49% and 3.58% for PointNet and PointNet2 respectively.

color information is particularly relevant to characterize urban objects, we fed the input points with 9-dimensional channels. In the training phase, we used the same fine tuning strategy than the one proposed in Section 5.4. Baseline models were first trained on vKITTI dataset for 50 epochs. We then concatenated our 3d-PSPNet at the end of the baseline models and fine tuned the whole network for another 50 epochs. In the control experiment, we continued training baseline models for 50 epochs.

Training and testing errors of PointNet and PointNet2 with and without our 3d-PSPNet module for the last 50 epochs are shown in Figure 9. The training curves of baseline models converge to a minimum after the first 50 training epochs. Testing curves with our 3d-PSPNet module oscillate in the first fine tuning 35 epochs but tend to convergence in the last 15 epochs. Our 3d-PSPNet increases the generalization of baseline models by improving OA by 2.49% and 3.58% for PointNet and PointNet2 respectively. Qualitative comparisons are illustrated in Figure 8. PointNet and PointNet2 produce accurate prediction results for most of the points. Our 3d-PSPNet module corrects errors for points belonging to large scale objects, i.e. *building* (red) and *car* (blue). Table 4 shows the 6-fold quantitative results on vKITTI dataset. Our 3d-PSPNet improves mIOU by 2.88% (respectively 3.95%) and mAcc by 3.27% (resp. 5.19%) with PointNet (resp. PointNet2) baseline. IOU of 13 and 11 out of 13 classes is improved for PointNet and PointNet2 baseline respectively. Both qualitative and quantita-

tive results show our 3d-PSPNet module improves accuracy by aggregating global contextual information from urban scene point clouds.

Method	mIOU	mAcc	terrain	tree	vegetation	building	road	guard rail	traffic sign	traffic light	pole	misc	truck	car	van
PointNet	28.43	38.65	54.19	84.43	19.43	29.16	59.77	12.27	20.31	2.81	10.74	1.98	8.82	44.39	21.29
PointNet + ours	**31.31**	**41.92**	**58.23**	**87.74**	**20.01**	**32.61**	**63.05**	**14.75**	**28.97**	**3.84**	**11.26**	**2.48**	**9.77**	**47.64**	**22.64**
PointNet2	30.94	40.09	56.41	81.32	24.94	27.07	58.34	19.99	25.10	**11.56**	12.54	1.40	**5.71**	54.62	23.25
PointNet2 + ours	**34.89**	**45.28**	**60.47**	**90.38**	**26.98**	**38.65**	**59.41**	**22.31**	**29.21**	8.89	**14.97**	**4.07**	5.68	**55.20**	**37.42**

Table 4: 6-fold quantitative results on the vKITTI dataset for PointNet and PointNet2, including mIOU, mAcc and IOU for 13 classes.

We also evaluated our methods on the Scene 6 of vKITTI dataset with DGCNN and PointSIFT basline models. The quantitative results are illustrated in Table 5. For DGCNN, our 3d-PSPNet improves mIOU by 4.63% and mAcc by 3.74%. Note that, for PointSIFT, the baseline model does not produce good results under the current experimental setting. In this case, our 3d-PSPNet module achieves 6.98% improvement for mIOU and 7.22% improvement for mAcc. This observation shows that our 3d-PSPNet succeeds in improving the generalization of baseline models by enriching pointwise features with multi-scale contextual clue when the local pointwise features are not sufficient.

Method	mIOU	mAcc	terrain	tree	vegetation	building	road	guard rail	traffic sign	traffic light	pole	misc	truck	car	van
DGCNN	39.48	49.15	88.64	86.51	0.00	13.81	81.67	58.45	31.68	0.00	24.78	0.00	0.00	69.58	**58.11**
DGCNN+Ours	**44.11**	**52.89**	**89.75**	**92.18**	0.00	**17.32**	**87.91**	**64.61**	**36.80**	0.00	**59.44**	0.00	0.00	**77.60**	47.91
PointSIFT	26.85	36.60	**81.03**	79.54	0.00	**15.43**	68.48	16.62	19.60	0.00	1.50	0.00	0.00	54.61	12.22
PointSIFT+Ours	**33.83**	**43.82**	76.57	**83.48**	0.00	8.53	**70.80**	**51.97**	**27.50**	0.00	**22.04**	0.00	0.00	**64.34**	**34.51**

Table 5: Quantitative results on Scene 6 of vKITTI dataset for DGCNN and PointSIFT, including mIOU, mAcc and IOU for 13 classes.

5.6. Network Architecture Design Analysis.

We analyzed the effect of some hyperparameters involved in our 3d-PSPNet module and discussed some design choices as well as the impact of input point features and the transfer learning strategy on the performances.

Number of pyramid scales L. We analyzed the impact of the number of pyramid scales L on the whole network. To do so, we selected $L \in \{1, 2, 3, 4, 5\}$ and performed 5 experiments under the same settings on S3DIS dataset using PointNet baseline with our 3d-PSPNet module. Figure 10 shows that L acts as a trade-off between prediction accuracy and efficiency. Increasing L gives better mIOU results but requires more training time. When L is smaller than 5, the training time and mIOU curves grow in a quasi-quadratic manner. However, when L is equal

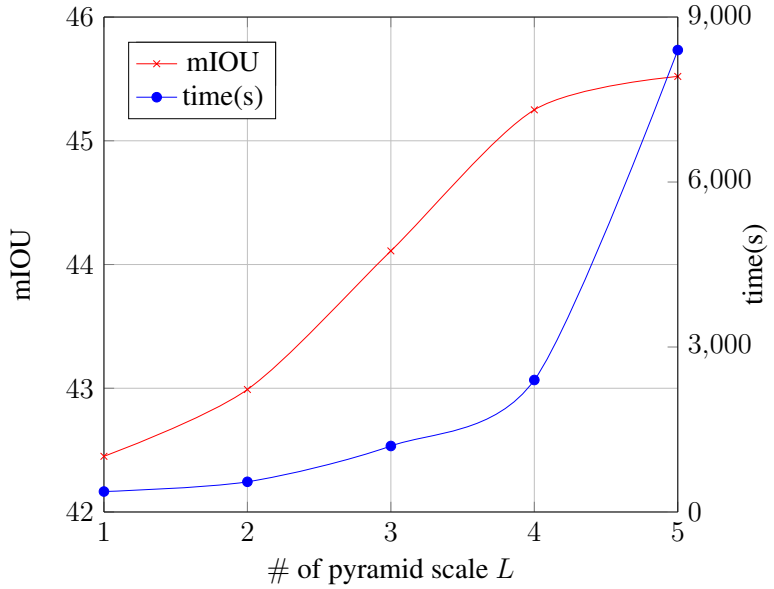


Figure 10: Analysis of *number of pyramid scales* L on prediction accuracy and efficiency. Noted that *time* refers to the average training time per epoch.

to 5, the computational time is very high but mIOU is marginally better. The explosion of computational time comes from the numerous operators imposed on the large number of grids ($16\times16\times16$). In our experiments, we used a 4-level pyramid structure. However, $L = 3$ is also a good choice for reducing training time.

Number of channels and size of MLP. We studied the impact of MLP on the accuracy and the computational cost of the network. Note that the only extra parameters to be trained in our 3d-PSPNet module were sequences of MLPs (fully connected layers) involved at each pyramid scale. These MLPs learn to aggregate information along the feature channel. To analysis the impact of MLPs on our framework, we performed 4 control experiments on S3DIS dataset with our 3d-PSPNet involving various MLPs. The segmentation results and model size are reported in Table 6. On one hand, increasing the output channels of fully connected layer slightly increases OA but also increases the complexity of the models to be learned. On the other hand, simply increasing the number of MLPs does not bring any benefit to the prediction accuracy and increases the model complexity. We believe that a more carefully designed composition of fully connected layers could achieve better prediction results. However, manually tuning the best MLP

architecture while bringing more parameters to be trained is a time-consuming task. Therefore, we impose a generic architecture by simply assigning a 256-dimensional fully connected layer in MLP at each pyramid scale.

MLP	OA(%)	mIOU(%)	Model size (MB)
128	81.16	44.34	**31.21**
256	81.28	**45.54**	40.23
512	**81.45**	45.46	58.27
[512,256,128]	80.87	44.34	56.84

Table 6: Study on *MLP*. Note that *model size* refers to the number of parameters to be learned in the whole framework.

Grid shape. As described in Section 4.1, there are two choices to divide the 3D scene space into sub-regional space. The first choice is the regular cubic grid which is used in our experiments. This structure is efficient to pool each point into its corresponding sub-regional grid by considering its spatial position in the 3D space. An alternative way is to employ the *sampling layer* of PointNet2 by selecting a subset of distant points and pool each original point into its corresponding distant point. This solution divides 3D scene into irregular grids according to the density of point clouds. We evaluated these two architectures on all three datasets with PointNet2 baseline under the same training settings. The evaluation results are given in Table 7. Our regular cubic grid version performs better on the different datasets. The main gap results from the invariance of density of regular cubic grids, where the pooling grid of each point is only dependent on its spatial coordinates.

Method	S3DIS Area 5	ScanNet	vKITTI Scene 6
Ours V1	**80.15**	**76.21**	**93.28**
Ours V2	78.79	75.81	93.09

Table 7: *Grid shape analysis*. V1 is the regular-grid method whereas V2 is the adaptive-grid method mentioned in Section 4.1. OA(%) is reported for both version.

Input features. We also analyzed the impact of the input feature channels. We separately fed input points with 3-dimensional channels $[x, y, z]$ and 9-dimensional channels $[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$ into both PointNet and PointNet2 models with and without our 3d-PSPNet module. We followed the control experiment setting proposed in Section 5.3. Table 8 presents a quantitative evaluation for four control experiments. Our 3d-PSPNet increases prediction accuracy in all pairs of control

experiments. However, when the input features are insufficient, i.e. when containing only spatial position $[x, y, z]$, our 3d-PSPNet performs better than by considering 9-dimensional features. The main reason is that pointwise features learned from rich input point features by baseline models are discriminative enough to reach accurate prediction results. In summary, our 3d-PSPNet module better preserves generalization for insufficient input features.

Method	$[x, y, z]$		$[x, y, z, r, g, b, \overline{x}, \overline{y}, \overline{z}]$	
	OA(%)	Improvement	OA(%)	Improvement
PointNet	77.12	2.20	79.49	1.79
PointNet+Ours	**79.32**		**81.28**	
PointNet2	76.37	3.78	83.11	1.54
PointNet2+Ours	**80.15**		**84.65**	

Table 8: *Input features analysis*.

Transfer learning. We studied how the transfer learning strategy helps accelerating the training procedure on the ScanNet and vKITTI datasets. In Section 5.4, we trained the baseline model for 20 epochs and fine-tuned the network equipped with our 3d-PSPNet module for another 20 epochs. In Section 5.5, we followed this strategy, used the pre-trained model as the staring point and fine tuned the whole network for another 50 epochs. To analyze the gain of the transfer learning strategy, we trained in the control experiments the PointNet2 baseline model equipped with our 3d-PSPNet module from scratch with random initialization. We followed the same experimental settings and trained the whole network for the same number of epochs, i.e. 40 and 100 respectively. Evaluation results are reported in Table 9. After the same number of epochs, transfer learning allows the whole network to converge to a better optimal than by training from scratch. In other words, transfer learning accelerates the training phase.

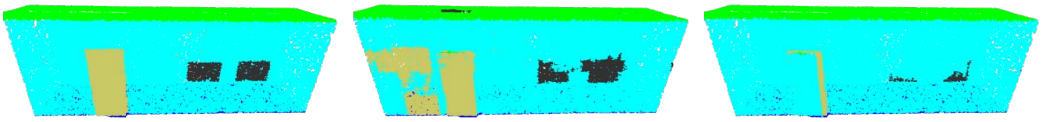
Method	ScanNet		vKITTI Scene 6	
	OA(%)	mIOU(%)	OA(%)	mIOU(%)
Learning from scratch	74.73	30.32	87.69	40.45
Fine-tuning	**76.21**	**33.02**	**93.28**	**45.48**

Table 9: *Learning from scratch* vs *transfer learning*.

5.7. Limitations

Our 3d-PSPNet module is designed to enrich local pointwise features with

contextual clues by interacting points at multiple scales. In case of simple scenes where the interactions between input points are trivial, our method does not necessarily improve the results of the baseline model. Figure 11 shows an example on the S3DIS dataset. Our fine-tuning strategy fails to correct the mislabeling points returned by the DGCNN baseline model. Meanwhile, the final network converges to an over-fitting state where correct predictions are mislabeled after using our module.



(a) Ground truth (b) DGCNN (c) DGCNN+Ours

Figure 11: Failure case. When the interactions between input points are simple, our module fails to correct the mislabeling results returned by the DGCNN baseline model, and can even degrades the overall accuracy (see the loss of *door* points in yellow and *clutter* points in black in the right image).

6. Conclusion

We proposed a pyramid structured network to aggregate multi-scale contextual information in point clouds. This generic module can be concatenated after any pointwise feature learning network. It enriches local features with multi-scale sub-regional global clues. Experimental results on different popular datasets demonstrated that the enriched pointwise features are more discriminative in complex 3D scenes and produce more accurate semantic segmentation predictions.

As perspective, we plan to design more efficient grid structures at different level of our pyramid network by adapting the decomposition of the 3D space to the spatial density of points. Moreover, we also want to extend this work to other tasks, such as 3D object detection and urban mesh semantic segmentation.

Acknowledgments

The authors thank CSTB for financial support and Sven Oesau for technical discussions.

References

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3d semantic parsing of large-scale indoor spaces, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Atzmon, M., Maron, H., Lipman, Y., 2018. Point convolutional neural networks by extension operators. arXiv preprint arXiv:1803.10091 .

Badrinarayanan, V., Kendall, A., Cipolla, R., 2015. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv preprint arXiv:1511.00561 .

Boulch, A., Guerry, J., Le Saux, B., Audebert, N., 2018. Snapnet: 3d point cloud semantic labeling with 2d deep segmentation networks. Computers & Graphics 71, 189–198.

Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2018. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence 40, 834–848.

Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M., 2017. Scannet: Richly-annotated 3d reconstructions of indoor scenes, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Dai, A., Nießner, M., 2018. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation. arXiv preprint arXiv:1803.10409 .

Deng, H., Birdal, T., Ilic, S., 2018a. Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors, in: Proc. of the European Conference on Computer Vision (ECCV).

Deng, H., Birdal, T., Ilic, S., 2018b. Ppfnet: Global context aware local features for robust 3d point matching, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Fang, H., Lafarge, F., Desbrun, M., 2018. Planar shape detection at structural scales, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Francis, E., Theodora, K., Alexander, H., Bastian, L., 2017. Exploring spatial context for 3d semantic segmentation of point clouds, in: Proc. of International Conference on Computer Vision (ICCV), Workshop.

Gojcic, Z., Zhou, C., Wegner, J.D., Wieser, A., 2018. The perfect match: 3d point cloud matching with smoothed densities. arXiv preprint arXiv:1811.06879 .

Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M., 2018. Atlasnet: A papier-mâché approach to learning 3d surface generation. arXiv preprint arXiv:1802.05384 .

Hackel, T., Savinov, N., Ladicky, L., Wegner, J.D., Schindler, K., Pollefeys, M., 2017. Semantic3d. net: A new large-scale point cloud classification benchmark. arXiv preprint arXiv:1704.03847 .

Huang, Q., Wang, W., Neumann, U., 2018. Recurrent slice networks for 3d segmentation of point clouds, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Jiang, M., Wu, Y., Lu, C., 2018. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. arXiv preprint arXiv:1807.00652 .

Kalogerakis, E., Averkiou, M., Maji, S., Chaudhuri, S., 2017. 3d shape segmentation with projective convolutional networks, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Klokov, R., Lempitsky, V., 2017. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models, in: Proc. of International Conference on Computer Vision (ICCV).

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Proc. of Advances in Neural Information Processing Systems (NIPS).

Lafarge, F., Mallet, C., 2012. Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation. International journal of computer vision 99, 69–85.

Landrieu, L., Simonovsky, M., 2018. Large-scale point cloud semantic segmentation with superpoint graphs, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Maturana, D., Scherer, S., 2015. Voxnet: A 3d convolutional neural network for real-time object recognition, in: Proc. of International Conference on Intelligent Robots and Systems (IROS).

Nurunnabi, A., Belton, D., West, G., 2012. Robust segmentation in laser scanning 3d point cloud data, in: Proc. of International Conference on Digital Image Computing Techniques and Applications (DICTA).

Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J., 2017a. Frustum pointnets for 3d object detection from rgb-d data. arXiv preprint arXiv:1711.08488 .

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017b. Pointnet: Deep learning on point sets for 3d classification and segmentation, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J., 2016. Volumetric and multi-view cnns for object classification on 3d data, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017c. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, in: Proc. of Advances in Neural Information Processing Systems (NIPS).

Rethage, D., Wald, J., Sturm, J., Navab, N., Tombari, F., 2018. Fully-convolutional point networks for large-scale point clouds, in: Proc. of the European Conference on Computer Vision (ECCV).

Riegler, G., Ulusoy, A.O., Geiger, A., 2017. Octnet: Learning deep 3d representations at high resolutions, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Rouhani, M., Lafarge, F., Alliez, P., 2017. Semantic segmentation of 3D textured meshes for urban scene analysis. ISPRS Journal of Photogrammetry and Remote Sensing 123.

Roynard, X., Deschaud, J.E., Goulette, F., 2018. Classification of point cloud scenes with multiscale voxel deep network. arXiv preprint arXiv:1804.03583 .

Simonovsky, M., Komodakis, N., 2017. Dynamic edge-conditioned filters in convolutional neural networks on graphs, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J., 2018. Splatnet: Sparse lattice networks for point cloud processing, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Su, H., Maji, S., Kalogerakis, E., Learned-Miller, E., 2015. Multi-view convolutional neural networks for 3d shape recognition, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Tchapmi, L., Choy, C., Armeni, I., Gwak, J., Savarese, S., 2017. Segcloud: Semantic segmentation of 3d point clouds, in: Proc. of International Conference on 3D Vision (3DV).

Wang, W., Yu, R., Huang, Q., Neumann, U., 2018a. Sgpn: Similarity group proposal network for 3d point cloud instance segmentation, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M., 2018b. Dynamic graph cnn for learning on point clouds. arXiv preprint arXiv:1801.07829 .

Weinmann, M., Jutzi, B., Mallet, C., 2013. Feature relevance assessment for the semantic interpretation of 3d point cloud data. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 5, W2.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3d shapenets: A deep representation for volumetric shapes, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y., 2018. Spidercnn: Deep learning on point sets with parameterized convolutional filters, in: Proc. of the European Conference on Computer Vision (ECCV).

Yu, L., Li, X., Fu, C.W., Cohen-Or, D., Heng, P.A., 2018. Pu-net: Point cloud upsampling network, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J., Funkhouser, T., 2017. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J., 2017. Pyramid scene parsing network, in: Proc. of Computer Vision and Pattern Recognition (CVPR).

Zhao, Y., Birdal, T., Deng, H., Tombari, F., 2018. 3d point-capsule networks. arXiv preprint arXiv:1812.10775 .

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.H., 2015. Conditional random fields as recurrent neural networks, in: Proc. of International Conference on Computer Vision (ICCV).