

# PointWeb: Enhancing Local Neighborhood Features for Point Cloud Processing

Hengshuang Zhao<sup>1\*</sup> Li Jiang<sup>1\*</sup> Chi-Wing Fu<sup>1</sup> Jiaya Jia<sup>1,2</sup>

<sup>1</sup>The Chinese University of Hong Kong <sup>2</sup>Tencent YouTu Lab

{hszhao, lijiang, cwf, leo{jia}@cse.cuhk.edu.hk

## Abstract

This paper presents PointWeb, a new approach to extract contextual features from local neighborhood in a point cloud. Unlike previous work, we densely connect each point with every other in a local neighborhood, aiming to specify feature of each point based on the local region characteristics for better representing the region. A novel module, namely Adaptive Feature Adjustment (AFA) module, is presented to find the interaction between points. For each local region, an impact map carrying element-wise impact between point pairs is applied to the feature difference map. Each feature is then pulled or pushed by other features in the same region according to the adaptively learned impact indicators. The adjusted features are well encoded with region information, and thus benefit the point cloud recognition tasks, such as point cloud segmentation and classification. Experimental results show that our model outperforms the state-of-the-arts on both semantic segmentation and shape classification datasets.

## 1. Introduction

We have witnessed great progress in image recognition tasks, such as image classification [11, 22, 26, 7, 9, 8] and semantic segmentation [14, 3, 35], which are mainly advanced by the development of deep learning techniques with massive model capacity. Beyond 2D image recognition, there is increasingly growing interest in 3D vision [18, 36, 6, 34, 4] for applications of autonomous driving, augmented reality, robotics, etc. Advent of large-scale high-res 3D datasets [1, 5] also gives rise to the environment of reasoning 3D data with deep neural networks.

Directly extending 2D image deep learning methods to 3D recognition tasks is not always feasible, since 3D scenes are usually described by a set of points that are unordered and scattered in 3D. It is also not reasonable to simply apply 2D feature aggregating operations like convolution to irregular point clouds, since these operations generally work on

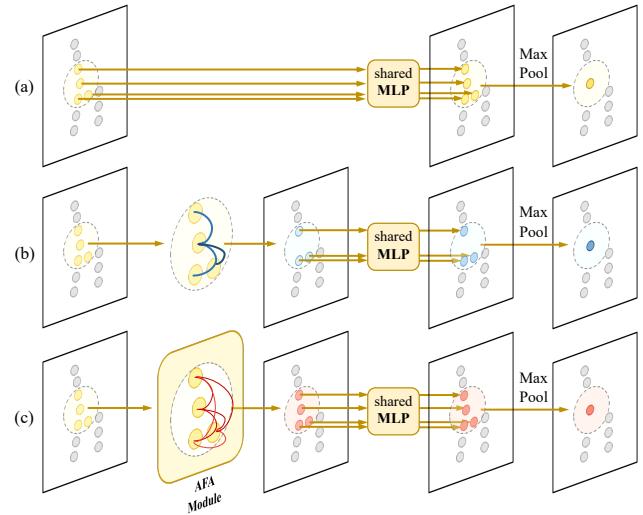


Figure 1. 3D segmentation illustration. (a) PointNet++ [20]. (b) DGCNN [32]. (c) Our approach with the Adaptive Feature Adjustment (AFA) module. Blue points represent the features that integrate pair information. Red points are the features that integrate region information. Compared with aggregating pair features from a center point to others by concatenation, our approach adaptively learns the impact indicator between each point pair, and aggregates features in the whole region.

regular grids. Approaches [16, 23, 21] addressed this problem by voxelizing point clouds and applying 3D CNN for feature learning, which is a natural thought. These methods run slowly and may suffer from information loss during the voxelization. Alternatively, PointNet [18] architecture directly processes raw point clouds with shared Multi-Layer Perception (MLP). The following PointNet++ [20] further improved the performance by introducing a hierarchical structure to extract global and local features. Unlike 2D convolutions that integrate features for a pixel with its local neighborhood, feature aggregation of a local region in PointNet++ is implemented by max pooling as in Fig. 1 (a).

For the Dynamic Graph CNN (DGCNN) [32], it aggregates information in each local region by concatenating features of a center point with the feature difference between

\*Equal contribution.

the center point and its  $k$  nearest neighbors, followed by an MLP and max pooling (Fig. 1 (b)). Here, the pair relationship is only considered for the center point and the consolidation of region information is still limited, since the actual operation for region aggregation is also achieved by a simple max pooling. PointCNN [13] addressed this problem by sorting the points into a potentially canonical order and applying convolution on the points. Whether points permuted and weighted by the  $\mathcal{X}$ -transformation [13] are in a canonical order or not yet requires further investigation.

Different from the above methods, we exhaust the contextual information in a local region by connecting and exploring all pairs of points in the region. To this end, we formulate a novel module, namely the Adaptive Feature Adjustment (AFA), to interconnect all pairs of points in the local region and eventually form a *locally fully-linked web*. We then learn the adjustment on the point feature from the point-pair difference as shown in Fig. 1 (c). This strategy enriches point feature in local region, and forms aggregated feature to better describe local region for 3D recognition.

Our network learns the impact indicators from the point-pair difference to determine feature adjustment, thus allowing an appropriate adaptation for better generality. Fig. 2 illustrates the process of feature adjustment. Further, we propose the PointWeb framework with our novel AFA module for 3D scene recognition, as shown in Fig. 4, which achieves state-of-the-art results on point cloud scene understanding tasks, including point cloud semantic segmentation and shape classification. The top ranking experimental results are achieved on the three most competitive datasets, *i.e.*, Stanford Large-Scale 3D Indoor Space (S3DIS) [1] for semantic segmentation, ScanNet [5] for semantic voxel labeling, and ModelNet40 [33] for shape classification. We believe this effective adaptive feature adjustment module can benefit other point cloud understanding tasks. We give all the implementation details, and make our code and trained models publicly available<sup>1</sup>. Our main contribution is twofold.

- We enable information interchange between each paired 3D points through adaptive feature adjustment (AFA). This module largely enhances the representation ability of the learned point-wise features.
- We propose the PointWeb framework with the key module AFA. It achieves top performance on various competitive point cloud datasets, thus demonstrating its effectiveness and generality.

## 2. Related Work

**3D Data Representation** Real scanned data has a collection of 3D point coordinates. To adapt the data for con-

volution, one straightforward approach is to voxelize it in a 3D grid structure [16, 23]. However, the representation is clearly inefficient, since most voxels are usually unoccupied. Later, OctNet [21] explored the sparsity of voxel data and alleviated this problem. However, the memory occupancy is still high when it comes to deeper neural networks. Moreover, since voxels are discrete representation of space, this method still requires high resolution grids with large memory consumption as a trade-off to keep a level of representation quality.

Another common 3D representation is in multi-view [19, 24, 25], where the point data is projected to various specific image planes in the 3D space to form 2D images. By this means, point data can be processed using conventional convolution on 2D images. This approach, however, ignores the intrinsic geometric relationship of 3D points, and the choice of image planes could heavily affect results. Occluded parts in the 3D data due to projection are not handled.

**Deep Learning on Point Cloud** PointNet [18] first discusses the irregular format and permutation invariance of point sets, and presents a network that directly consumes point clouds. PointNet++ [20] extends PointNet by further considering not only the global information but also the local details with a farthest sampling layer and a grouping layer. Although the local context is exploited in PointNet++, information in local regions may not be well aggregated using only max pooling. Hence, DGCNN [32] aggregates the local context information by linking each center point with its  $k$  nearest neighbors. The paired features are then independently encoded by an MLP. The aggregation operation on the local regions is still a simple max pooling.

Recent methods improve context integration by extending the convolution operator to regular grid structure to handle unordered points. PCCN [31] has parametric continuous convolution operations that define a kernel function over the continuous support domain. PointCNN [13] exploits the canonical order of points by permuting and weighting input points and features with a  $\mathcal{X}$ -Conv operator, the re-organized points are then processed by a conventional convolution. Besides, Superpoint Graph (SPG) [12], on the other hand, focuses on dealing with large point clouds. The points are adaptively partitioned into geometrically homogeneous elements to build a superpoint graph, which is then fed into a graph neural network for producing the semantic labels.

Our work also centers on the aggregation of local features. Unlike previous methods that adapt convolution in point clouds, we put our attention onto the interaction between points in each local neighborhood region. By exhausting the context information between all point pairs, our network module refines the features to make them more descriptive regarding the local neighborhood region.

<sup>1</sup><https://github.com/hszhao/PointWeb>

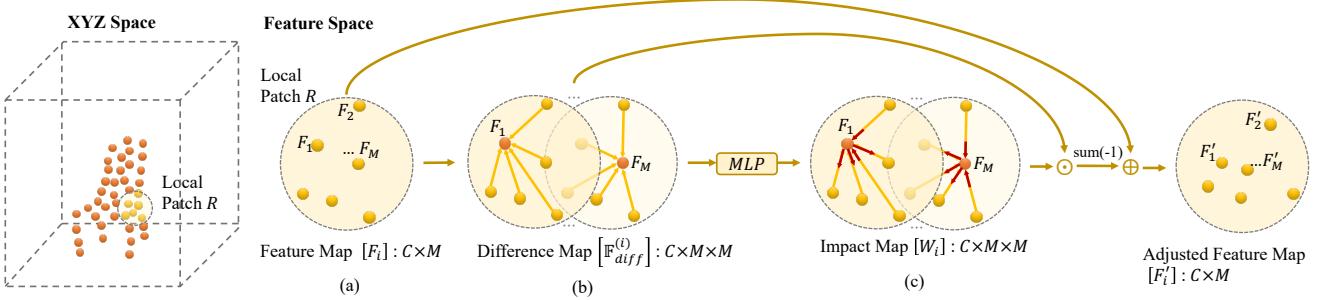


Figure 2. Our Adaptive Feature Adjustment (AFA) module.

### 3. Our Method

Exploring the relationship among points in a local region is the focus of this paper. Particularly, after we extract pointwise features (or point features, for short) using a neural network, further aggregating these local features helps improve the point cloud recognition quality for the tasks of semantic segmentation and classification.

Given a 3D point cloud, PointNet++ [20] uses the farthest point sampling to choose points as centroids, and then applies kNN to find the neighboring points around each centroid, which well defines the local patches in the point cloud. For a local patch (or local neighborhood)  $R$  of  $M$  points, we denote by  $\mathbb{F}$  the set of point features in  $R$ , such that  $\mathbb{F} = \{F_1, F_2, \dots, F_M\}$ , where  $F_i \in \mathbb{R}^C$ .  $C$  denotes the number of channels in each point feature.

Here, the ultimate goal is to obtain a representative feature  $F_{out} \in \mathbb{R}^{C_{out}}$  of region  $R$ , where  $C_{out}$  is the number of channels in the output feature. PointNet++ obtains the representative feature using MLP followed by max pooling. However, the procedure does not involve regional information exchange among the points in local neighborhood.

In our approach, we densely connect points in  $R$  as a *local web of points*, and formulate an *Adaptive Feature Adjustment* (AFA) module to learn the impact of each point on other points for adjusting their features. By this means, we take the neighborhood context into the point features and enhance the capability of the features to describe the local neighborhood. Fig. 2 gives an overview of the AFA module. We call our overall network *PointWeb*, since our approach effectively extracts local neighborhood context through a web of densely-connected points.

#### 3.1. Adaptive Feature Adjustment (AFA) Module

Given region  $R$  and its feature set  $\mathbb{F} = \{F_1, F_2, \dots, F_M\}$ , we first formulate the adaptive feature adjustment (AFA) module to enhance the point features in  $\mathbb{F}$  by learning the contextual information in local neighborhood as

$$F'_i = F_i + \Delta F_i \text{ and } \Delta F_i = f_{mod}(F_i, \mathbb{F}), \forall F_i \in \mathbb{F}, \quad (1)$$

where  $F'_i$  is the enhanced  $F_i$ , and  $\Delta F_i$  is learned from  $\mathbb{F}$  through the feature modulator denoted as  $f_{mod}$ .

The next challenge is to formulate the feature modulator to efficiently exchange and further aggregate information in  $\mathbb{F}$ . Intuitively, different features in the local region impose varied impact to enhance every  $F_i$ . Our feature modulator addresses this problem by adaptively learning the amount of impact given by each feature in  $\mathbb{F}$  on each  $F_i$ . It is expressed as

$$f_{mod}(F_i, \mathbb{F}) = \sum_{j=1}^M f_{imp}(F_i, F_j) \cdot f_{rel}(F_i, F_j), \quad (2)$$

where  $f_{imp}$  is a function that is learned for calculating the amount of impact of  $F_j$  on  $F_i$ , while  $f_{rel}$  represents how  $F_j$  relates to  $F_i$ . It is worth noting that we also include the self-impact of  $F_i$  in the modulator.

##### 3.1.1 Impact Function $f_{imp}$

The Multi-Layer Perception (MLP) network was presented in [18, 20] that approximates a general function over a point set. We use MLP to calculate the impact function  $f_{imp}$ , as illustrated in Fig. 3. It is formulated as

$$w_{ij} = f_{imp}(F_i, F_j) = \text{MLP}(g(F_i, F_j)), \quad (3)$$

where  $g$  is a function to combine features  $F_i$  and  $F_j$ , and  $w_{ij}$  is the resulting impact indicator of  $F_j$  on  $F_i$ .

One simple approach to model  $g$  is to just concatenate the two features. This solution has an obvious limitation that  $g$  completely contains  $F_i$  and  $F_j$ , where half of the feature channels remain unchanged even if  $F_j$  varies. This makes  $F_i$  dominate when calculating the impact. Another choice is to take the feature sum ( $F_i + F_j$ ) as  $g$ . We note this strategy is also problematic since the impact of  $F_j$  on  $F_i$  becomes the same as that of  $F_i$  on  $F_j$ . This type of *symmetric* impact yields an undesirable property, which will be demonstrated experimentally later.

With these considerations, we thus model  $g(F_i, F_j) = F_i - F_j$ , making the impact calculated as the difference between the two feature vectors. We will show later in Table 3

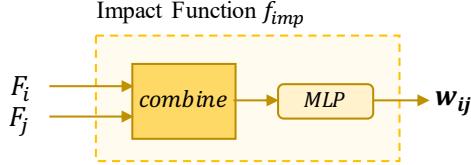


Figure 3. Illustration of the impact function  $f_{imp}$  for  $F_j$  on  $F_i$ .

the statistics from experiments of using the three different forms of  $g$ . Note here  $i = j$  is a special case, for which we set  $g(F_i, F_i)$  as  $F_i$ . Therefore, the impact of  $F_i$  on itself is estimated by its own feature  $F_i$ .

### 3.1.2 Relation Function $f_{rel}$

On the other hand, the relation function  $f_{rel}$  aims to decide how the impact indicator  $w_{ij}$  acts on  $F_i$ . A naive method is to directly multiply  $w_{ij}$  with  $F_j$  as

$$f_{rel}(F_i, F_j) = F_j. \quad (4)$$

Then, the overall  $f_{mod}$  in Eq. (2) becomes

$$f_{mod}(F_i, \mathbb{F}) = \sum_{j=1}^M f_{imp}(F_i, F_j) \cdot F_j. \quad (5)$$

Though the result quality of point cloud recognition tasks using this naive relation function already improves compared to the baseline, the performance of our framework can be further boosted using a different vector form for  $f_{rel}$ . Mathematically, we model the relation function as

$$f_{rel}(F_i, F_j) = \begin{cases} F_i - F_j & \text{if } i \neq j \\ F_i & \text{if } i = j \end{cases}. \quad (6)$$

Please refer to the ablation study in Section 4 to compare the performance using different forms of relation functions.

Now, for each feature  $F_i$  in local region  $R$ , the overall output of the feature adjustment is

$$F'_i = \alpha_i^{(i)} \cdot F_i + \sum_{j=1, j \neq i}^M \alpha_j^{(i)} \cdot (F_j - F_i), \quad (7)$$

where

$$\alpha_j^{(i)} = \begin{cases} -f_{imp}(F_i, F_j) & \text{if } i \neq j \\ 1 + f_{imp}(F_i, F_i) & \text{if } i = j \end{cases}. \quad (8)$$

In other words, the formulation works like a force field in the local region  $R$  (see Fig. 2 (c)), where every other feature in  $R$  acts on  $F_i$  with a force (in the feature space), trying to push  $F_i$  towards or away from itself. The intensity and direction of the force are determined by the coefficient  $\alpha_j^{(i)}$ , which is adaptively learned according to the difference between the two feature vectors. Hence, the output  $F'_i$  incorporates the context information of the whole region, thus better describing the characteristics of the region.

### 3.1.3 Element-wise Impact Map

Besides the two key functions  $f_{imp}$  and  $f_{rel}$ , it is noted that the impact factors  $w_{ij} = f_{imp}(F_i, F_j)$  ( $j = 1, \dots, M$ ) operate on the feature difference map in an element-wise fashion. The length of each factor equals the number of channels in the feature. Considering that the interaction between two point features may differ throughout the channels, instead of computing point-wise impact factors, an element-wise impact map covering the whole region and all channels are obtained for each feature in the local region. The impact map for  $F_i$  is formulated as

$$\mathbf{W}_i = [w_{i1}, w_{i2}, \dots, w_{iM}]. \quad (9)$$

Denote the number of channels as  $C$  and the size of the impact map as  $C \times M$ . The feature modulator is then represented in matrix form as

$$f_{mod}(F_i, \mathbb{F}) = (\mathbf{W}_i \odot \mathbb{F}_{diff}^{(i)})e, \quad (10)$$

where  $\odot$  denotes element-wise multiplication,  $e$  is an all-one vector, and  $\mathbb{F}_{diff}^{(i)}$  is the feature difference map of size  $C \times M$ . Specifically,

$$\mathbb{F}_{diff}^{(i)} = [F_1 - F_i, \dots, F_i, \dots, F_M - F_i]. \quad (11)$$

## 3.2. PointWeb with Local Feature Modulation

Our framework is built upon the PointNet++ architecture, which is a hierarchical network composed of several set-abstraction levels. Inside each local region in the abstraction module, both the global and local features are incorporated for point set recognition. However, in PointNet++, the only operation to aggregate information in each local region is max pooling, while our *PointWeb* framework builds a fully-linked web of points among the point features in each local region and then integrates the information of the region by learning to determine their mutual influence. The overall framework of PointWeb for semantic segmentation is shown in Fig. 4. The major ingredient for improving the performance is the Adaptive Feature Adjustment (AFA) module, which is the highlighted box in the figure.

In detail, each grouping layer is followed by an AFA module introduced in Section 3.1. Please refer to Fig. 2 for the illustration of the AFA module for a local region. Specifically, every two features in the same local region are linked to form a web (Fig. 2 (b)). Then, the features are updated by considering the interactions among them (Fig. 2 (c)). This framework makes the same feature grouped in different local regions specified on the basis of the region characteristics.

Without changing the number and size of features, our AFA module can be seen as a feature transformation module, which exchanges and aggregates context information

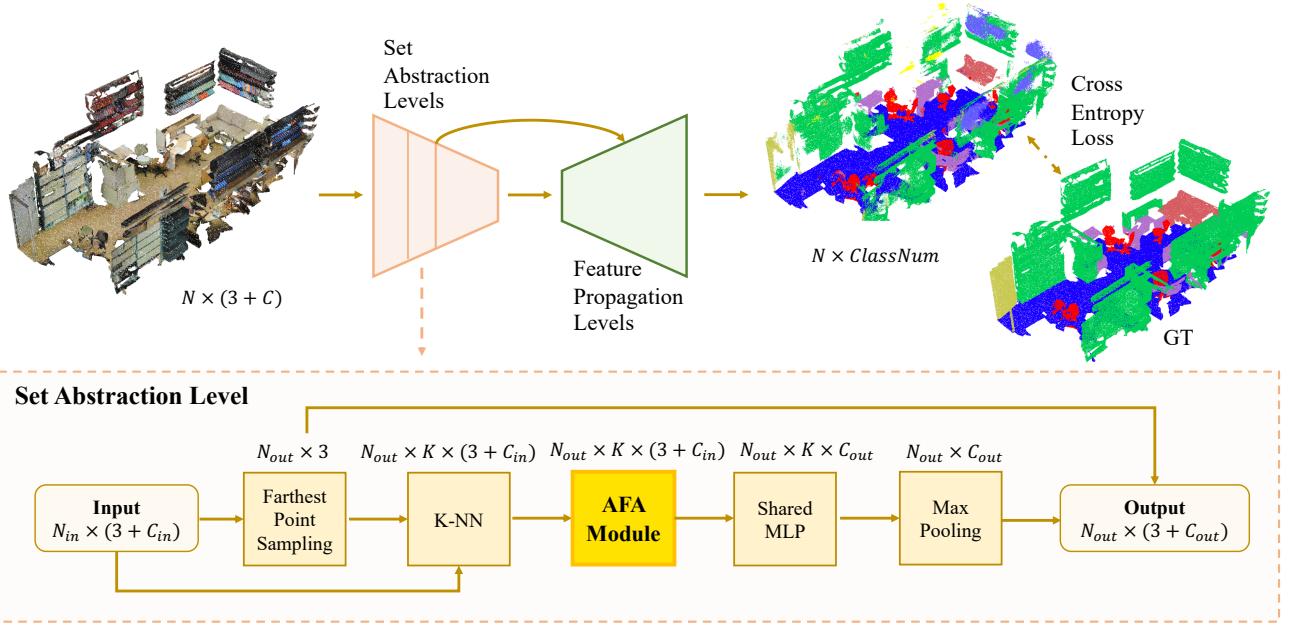


Figure 4. Architecture of our PointWeb for point cloud semantic segmentation. Adaptive Feature Adjustment (AFA) module follows the kNN grouping layer for constructing a fully-connected web in each local region and transferring information among points.

throughout the space and channel in each local region. The adjusted features are then given to MLP and max pooling for further information integration in different channels.

## 4. Experimental Evaluation

Our proposed PointWeb framework is effective for point cloud scene understanding. To demonstrate the effectiveness, we conduct experiments on both point cloud semantic segmentation and classification tasks. Two large-scale 3D point cloud segmentation datasets, including Stanford Large-Scale 3D Indoor Space (S3DIS) [1] and ScanNet [5], are adopted. Another shape classification dataset ModelNet [33] is used for classification evaluation.

### 4.1. Implementation Details

We conduct our experiments based on the PyTorch [17] platform. During the training, we use the SGD solver with base learning rate of 0.05 and a mini-batch size of 16. Momentum and weight decay are set to 0.9 and 0.0001, respectively. For the S3DIS dataset, we train for 100 epochs and decay the learning rate by 0.1 for every 25 epochs. For the ScanNet and ModelNet40 datasets, we train for 200 epochs and decay the learning rate by 0.1 for every 50 epochs.

### 4.2. S3DIS Semantic Segmentation

**Data and Metric** The S3DIS [1] dataset contains 3D scans in six areas including 271 rooms. Each point in the scan is annotated with one of the semantic labels from 13

categories (chair, table, ceiling, floor, clutter etc.). To prepare the training data, we follow [20], where the points are uniformly sampled into blocks of area size  $1m \times 1m$ . Each point is represented with a 9D vector ( $XYZ$ , RGB and a normalized location in the room).

During the training, we randomly sample 4,096 points from each block on-the-fly. During the testing, we adopt all the points for evaluation. Following [20, 28, 12, 13], we report the results on two settings, *i.e.*, testing on Area 5 (rooms are not present in other folds) and 6-fold cross validation (calculating the metrics with results from different folds merged). For the evaluation metrics, we use mean of *class-wise intersection over union* (mIoU), mean of *class-wise accuracy* (mAcc), and overall *point-wise accuracy* (OA).

**Performance Comparison** Tables 1 and 2 show the quantitative results of different methods under the two settings mentioned above. In this highly competitive dataset, our PointWeb achieves the highest performance in terms of mIoU and OA on Area 5 evaluation and yields the highest mIoU and mAcc on the 6-fold setting. The mIoU of PointWeb reaches 60.28% on Area 5, 2.01% higher than the current state-of-the-art, PCCN. Meanwhile, the mIoU of PointWeb on the 6-fold cross evaluation reaches 66.73%, outperforming the previous best method by 1.34 points.

Visual demonstration is given in Fig. 5. PointWeb well captures certain detailed structures in the point clouds. As shown in the figure, inconspicuous object parts, like legs and table, can be distinguished and recognized cor-

Method	OA	mAcc	mIoU	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [18]	-	48.98	41.09	88.80	97.33	69.80	0.05	3.92	46.26	10.76	58.93	52.61	5.85	40.28	26.38	33.22
SegCloud [28]	-	57.35	48.92	90.06	96.05	69.86	0.00	18.37	38.35	23.12	70.40	75.89	40.88	58.42	12.96	41.60
PointCNN [13]	85.91	63.86	57.26	92.31	98.24	79.41	0.00	17.60	22.77	62.09	74.39	80.59	31.67	66.67	62.05	56.74
SPGraph [12]	86.38	66.50	58.04	89.35	96.87	78.12	0.00	42.81	48.93	61.58	84.66	75.41	69.84	52.60	2.10	52.22
PCCN [31]	-	<b>67.01</b>	58.27	92.26	96.20	75.89	0.27	5.98	69.49	63.45	66.87	65.63	47.28	68.91	59.10	46.22
PointWeb	<b>86.97</b>	66.64	<b>60.28</b>	91.95	98.48	79.39	0.00	21.11	59.72	34.81	76.33	88.27	46.89	69.30	64.91	52.46

Table 1. Semantic segmentation results on S3DIS dataset evaluated on Area 5.

Method	OA	mAcc	mIoU	ceiling	floor	wall	beam	column	window	door	table	chair	sofa	bookcase	board	clutter
PointNet [18]	78.5	66.2	47.6	88.0	88.7	69.3	42.4	23.1	47.5	51.6	54.1	42.0	9.6	38.2	29.4	35.2
RSNet [10]	-	66.45	56.47	92.48	92.83	78.56	32.75	34.37	51.62	68.11	60.13	59.72	50.22	16.42	44.85	52.03
SPGraph [12]	85.5	73.0	62.1	89.9	95.1	76.4	62.8	47.1	55.3	68.4	73.5	69.2	63.2	45.9	8.7	52.9
PointCNN [13]	<b>88.14</b>	75.61	65.39	94.78	97.3	75.82	63.25	51.71	58.38	57.18	71.63	69.12	39.08	61.15	52.19	58.59
PointWeb	87.31	<b>76.19</b>	<b>66.73</b>	93.54	94.21	80.84	52.44	41.33	64.89	68.13	71.35	67.05	50.34	62.68	62.20	58.49

Table 2. Semantic segmentation results on S3DIS dataset with 6-folds cross validation.

Method	mIoU	mAcc	OA
Baseline	55.88	62.30	85.29
No-Link	55.12	61.55	85.89
Summation	53.07	59.87	84.25
Subtraction	<b>60.28</b>	<b>66.64</b>	<b>86.97</b>
Concatenation	58.65	65.41	86.22

Table 3. Ablation results on the S3DIS dataset Area 5.

Method	mIoU	mAcc	OA
GANN	58.23	65.15	86.10
PointWeb	<b>60.28</b>	<b>66.64</b>	<b>86.97</b>
with Softmax	59.11	65.79	86.44
with Channel Share	58.27	64.99	86.22

Table 4. Compared with GANN [29] on the S3DIS dataset Area 5.

rectly. The last row shows two failure cases. In the left case, part of clutter on the wall is misclassified as board. While in the second one, clutter-like part on the wall is recognized in our algorithm but not in the ground truth. Part of the clutter in the left is mixed with the bookcase.

**Ablation Study** To better understand the influence of our design logic, we conduct ablation experiments on the S3DIS dataset adopting Area 5 for evaluation. We first investigate the combination function  $g$  in  $f_{imp}$  inside each local group with four different styles, *i.e.*, no combination between each paired points, combination through feature summation, feature subtraction, and feature concatenation. The quantitative results are shown in Table 3. Our choice of subtraction operation on local feature for adjustment is more discriminative and representative than the other alternatives.

Learning adjustment on individual feature without interaction with other features fails to harvest the local region context. Summation operation of paired features results in

a decrease of discriminative ability of features in local regions. It thus harms the recognition ability. Concatenation of paired features makes part of the feature representation almost the same, which is not as good as our choice of subtraction. With our subtraction operation, the combination between each feature pair  $F_i$  and  $F_j$  is unique and thus enhances the classification ability.

Further, we compare with the Graph Attention Network (GANN) [29] that is designed for graph problems. GANN builds connections with concatenation. Softmax operation is applied on the attention coefficients whereas AFA does not. Hence, the impact of AFA can be positive or negative, acting as push or pull. Also, our impact indicators vary over channels as represented in Eq. (9), giving higher flexibility than the scale coefficients in GANN that share weights across channels. The comparisons are listed in Table 4 where our PointWeb gets the highest scores.

Last but not least, for subtraction operation, we investigate the self-adaptive learned impact indicators operated on the feature itself directly other than working on the feature difference map alternatively. With impact factors imposed directly on the features ( $f_{rel}(F_i, F_j) = F_j$ ), the testing result is 59.79%/66.63%/86.51% (for mIoU/mAcc/OA). It is not as good as the impact feature that operates on the difference maps ( $f_{rel}(F_i, F_j) = F_i - F_j$ ), which yields the highest performance as 60.28/66.64/86.97 (%).

**Feature Visualization** For full understanding of the proposed AFA module, we produce the T-SNE [15] visualization on the input and output features that are with or without the adjustment module processing. Features on S3DIS with 13 categories are used. We randomly sample 100 points from the full area for each category and plot the feature distribution as illustrated in Fig. 7. From left to right are the input features, output features generated by the baseline net-

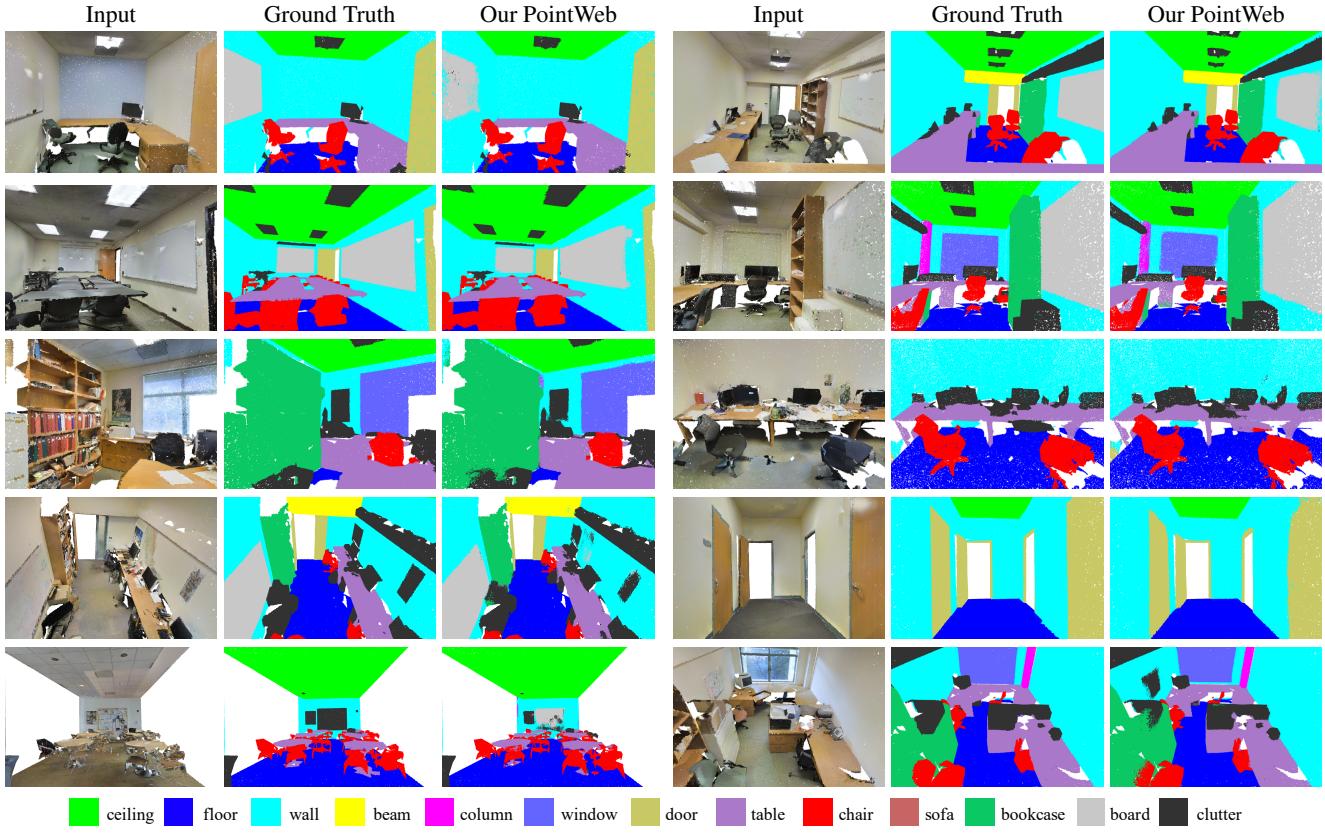


Figure 5. Visualization of semantic segmentation results on Stanford 3D Indoor Space dataset.

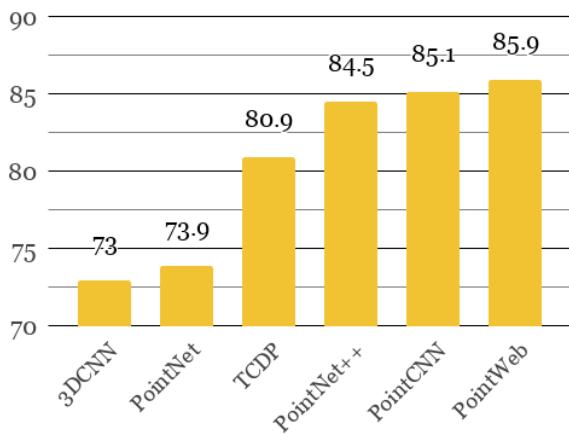


Figure 6. Semantic voxel labeling accuracy on ScanNet dataset. Compared approaches are 3DCNN [2], PointNet [18], TCDP [27], PointNet++ [20], PointCNN [13], and our PointWeb.

work without AFA and output feature by our PointWeb architecture with the AFA module to enhance the information exchange and connect paired points inside local regions.

Compared with the baseline network, where the local region features are processed independently with just MLPs,

our densely-connected feature adaption can promote the network learning to generate more compact, representative and more distinctive feature representations. Thus points of different categories are recognized more easily and precisely. The qualitative visualization indicates that the AFA module clearly enhances the discriminative ability of point features.

### 4.3. ScanNet Semantic Voxel Labeling

The ScanNet [5] dataset contains 1,513 scanned and reconstructed indoor scenes, split into 1201/312 for training and testing. For the semantic voxel labeling task, 20 categories are used for evaluation and 1 class for free space. We follow previous data processing pipeline [5, 20], where points are uniformly sampled from scenes and are divided into blocks, each of size  $1.5\text{m} \times 1.5\text{m}$ . During the training, 8,192 point samples are chosen, where no less than 2% voxels are occupied and at least 70% of the surface voxels have valid annotation. Points are sampled on-the-fly. All points in the testing set are used for evaluation and a smaller sampling stride of 0.5 between each pair of adjacent blocks is adopted during the testing. In the evaluation, overall *semantic voxel labeling accuracy* is adopted. For fair comparisons with the previous approaches, we do not use the RGB color

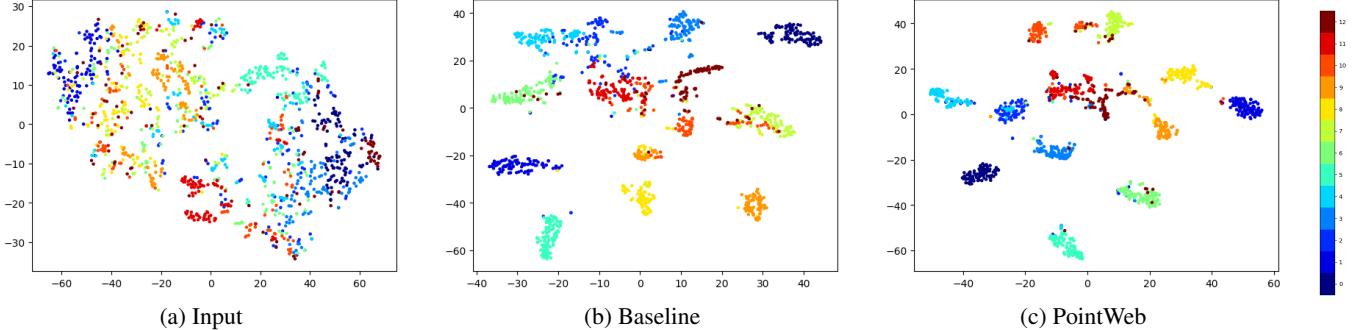


Figure 7. T-SNE visualization of features on S3DIS datasets with 13 classes: (a) input feature, (b) baseline output feature without AFA module (c) PointWeb output feature with AFA module incorporated.

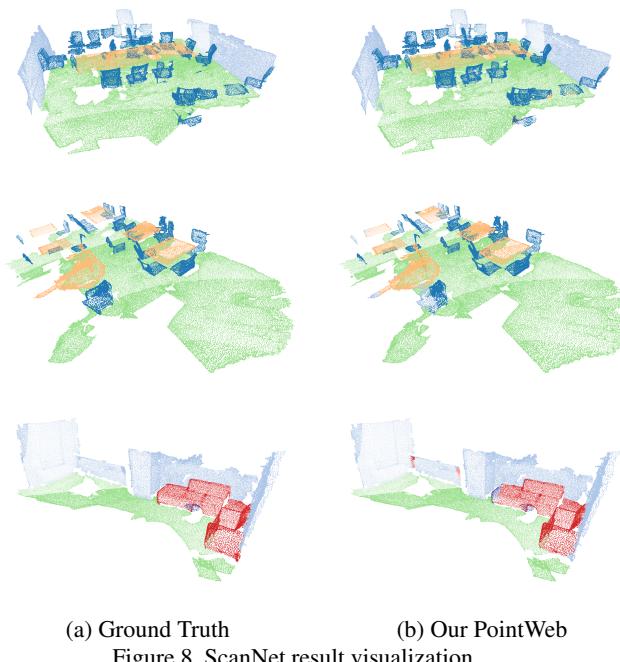


Figure 8. ScanNet result visualization.

information for training and testing.

Fig. 6 shows the semantic voxel labeling results. Our method reaches the first place on this dataset. Visual results are shown in Fig. 8, where PointWeb well classifies the surrounding objects and stuff in various rooms.

#### 4.4. ModelNet40 Classification

The ModelNet40 [33] dataset contains 12,311 CAD models from 40 object categories. They are split, so that 9,843 models are used for training and 2,468 for testing. Following [20], we uniformly sample the points from each CAD model together with computed normal vectors from the object meshes. For the classification framework, we replace the feature propagation layers with global max pooling and fully connected (FC) layers. Points in each shape model are randomly shuffled to augment the training set.

Method	input	mAcc	OA
3DShapeNets [33]	voxel	77.3	84.7
VoxNet [16]	voxel	83.0	85.9
Subvolume [19]	voxel	86.0	89.2
MVCNN [24]	image	-	90.1
PointNet [18]	point	86.2	89.2
PointNet++ [20]	point	-	91.9
SpecGCN [30]	point	-	92.1
DGCNN [32]	point	90.2	92.2
PointCNN [13]	point	88.1	92.2
PointWeb	point	89.4	92.3

Table 5. Shape classification results on ModelNet40 dataset.

Two dropout layers with drop rate 0.5 are added into the last two FC layers to reduce the over-fitting.

Table 5 shows the results of different methods evaluated regarding the overall accuracy and mean accuracy in each category. Our method is one of the top ranking solutions on this dataset. It shows a principled way to improve the applicability of point-based strategies for point cloud understanding.

## 5. Concluding Remarks

We have presented the adaptive feature adjustment (AFA) module and PointWeb architecture for 3D point cloud processing and recognition. It sets a dense connection of point pairs inside local regions, thus enabling every point to gather features from all others. Compared with the traditional approaches that are not aware of the local context and information interchange with other components, our framework enables better learning of feature representations for point cloud processing. Extensive experiments with our state-of-the-art results on three competitive datasets demonstrate the effectiveness and generality of our approach. We believe the proposed module can, in principle, advance the research of 3D scene understanding in the community.

## References

- [1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D semantic parsing of large-scale indoor spaces. In *CVPR*, 2016.
- [2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. In *ICLR*, 2014.
- [3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2018.
- [4] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016.
- [5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *CVPR*, 2017.
- [6] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *CVPR*, 2018.
- [7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [8] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.
- [9] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *CVPR*, 2017.
- [10] Qiangui Huang, Weiyue Wang, and Ulrich Neumann. Recurrent slice networks for 3d segmentation of point clouds. In *CVPR*, 2018.
- [11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.
- [12] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *CVPR*, 2018.
- [13] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhua Di, and Baoquan Chen. Pointcnn: Convolution on  $\mathcal{X}$ -transformed points. In *NIPS*, 2018.
- [14] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.
- [15] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 2008.
- [16] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 2015.
- [17] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS Workshop*, 2017.
- [18] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.
- [19] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, 2016.
- [20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.
- [21] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *CVPR*, 2017.
- [22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.
- [23] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017.
- [24] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015.
- [25] Hao Su, Fan Wang, Eric Yi, and Leonidas J. Guibas. 3d-assisted feature synthesis for novel views of an object. *ICCV*, 2015.
- [26] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.
- [27] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3D. *CVPR*, 2018.
- [28] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *3DV*, 2017.
- [29] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.
- [30] Chu Wang, Babak Samari, and Kaleem Siddiqi. Local spectral graph convolution for point set feature learning. In *ECCV*, 2018.
- [31] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *CVPR*, 2018.
- [32] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph cnn for learning on point clouds. *arXiv:1801.07829*, 2018.
- [33] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Liguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.
- [34] Li Xu and Jiaya Jia. Stereo matching: An outlier confidence approach. In *ECCV*, 2008.
- [35] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017.
- [36] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018.