Pacific Graphics (2018)
H. Fu, A. Ghosh, and J. Kopf (Editors)

Short Paper

Progressive 3D Scene Understanding with Stacked Neural Networks

Y. Song and Z. Sun

Nanjing University, State Key Laboratory for Novel Software Technology, China

Abstract

3D scene understanding is difficult due to the natural hierarchical structures and complicated contextual relationships in the 3d scenes. In this paper, a progressive 3D scene understanding method is proposed. The scene understanding task is decomposed into several different but related tasks, and semantic objects are progressively separated from coarse to fine. It is achieved by stacking multiple segmentation networks. The former network segments the 3D scene at a coarser level and passes the result as context to the latter one for a finer-grained segmentation. For the network training, we build a connection graph (vertices indicating objects and edges' weights indicating contact area between objects), and calculate a maximum spanning tree to generate coarse-to-fine labels. Then we train the stacked network by hierarchical supervision based on the generated coarse-to-fine labels. Finally, using the trained model, we can not only obtain better segmentation accuracy at the finest-grained than directly using the segmentation network, but also obtain a hierarchical understanding result of the 3d scene as a bonus.

CCS Concepts

•***Computing methodologies*** → ***Scene understanding; Neural networks;*** *Shape representations;*

1. Introduction

3D scene understanding is an important issue in computer vision and plays a fundamental role in many applications, such as robotics, augmented reality, autopilot, etc. [NL13]. 3D scene understanding is usually achieved through semantic segmentation, with the objective of segmenting and labeling the 3d scene into several semantic objects.

In recent years, due to the excellent generalization ability of neural networks, 3D scene segmentation method based on neural networks [LLZ*17, QLJ*17, TCA*17, QSMG16, QYSG17] has gradually become the mainstream. The basic structure of these methods is composed of a convolution or convolution-like module for extracting point or voxel features and a prediction module for point or voxel classification. However, there are complex contextual relationships among objects, and complex hierarchical structures in the scene, making it difficult for these methods to understand 3D scenes accurately. For the better use of context, we use prior knowledge of semantic parts as additional input [LZ17, ZSQ*17]. For the better use of hierarchy, we stack neural networks, and use hierarchical supervision to ensure that the model learns the hierarchical structure [ZYY*16]. An alternative strategy can be introduced to understand 3D scene progressively with refined semantic classes to incorporate object context. Before the final fine-grained segmentation, a scene is first segmented coarsely discarding the small objects to provide coarse-grained context. This can be achieved by stacking several segmentation networks with the former one segmenting coarsely and the latter one segmenting finely, while the former result is transmitted to the latter network as additional context prior.

In this paper, a progressive 3D scene understanding method by stacking neural networks is proposed. The 3D scene understanding task is decomposed into several different but related tasks, and semantic objects are progressively separated from coarse to fine in accordance with the contact relationship between objects. In order to achieve this progressive 3D scene understanding, we stack multiple segmentation networks. The result of the former network is passed to the latter one as context information. For the network training, we build a connection graph (vertices indicating objects and edges' weights indicating contact areas between objects), and calculate a maximum spanning tree to generate coarse-to-fine labels. Then we train the stacked network by hierarchical supervision based on the generated coarse-to-fine labels. Finally, using the trained model, we can not only obtain better segmentation accuracy at the finest-grained than directly using the segmentation network, but also obtain a hierarchical understanding results of the 3d scene as a bonus.

The main contributions of our work are as follows:

- We propose a stacking framework for progressive 3D scene understanding. Three base networks for 3D scene segmentation are stacked one after another, with the output of the former network combined with the raw input feeding to the latter one as additional context prior.
- We propose to generate hierarchical labels by applying a maximum spanning tree algorithm on the object connection graph, and train the stacked network by hierarchical supervision based on the generated coarse-to-fine labels.

DOI: 10.2312/pg.20181280

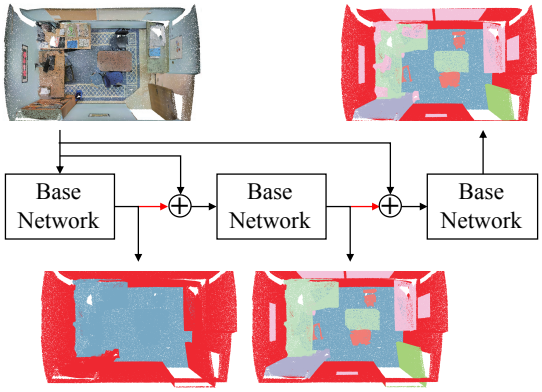

Figure 1: *Network architecture. PointNet++ [QYSG17] is used as our base network, it takes point cloud as input and outputs point-wise labels as segmentation result. Three base networks with are stacked one after anther to form the stacked network. They take point cloud as inputs and output segmentation results at different levels from coarse to fine. Cross-layer skip links (red arrows) concatenate the coarser-grained network output to the finer-grained network input to provide context prior.*

2. Stacked networks for scene understanding

According to the principle of totality in the gestalt psychology, humans usually pay attention to the overall layout before focusing on the specific objects when observing an unfamiliar environment. Inspired by this coarse-to-fine understanding process, we can use several networks to segment 3D scene progressively with refined semantic classes for 3D scene understanding task.

2.1. Base network

The base network is a segmentation network handling 3D point clouds or voxels. In order to simplify the expression, we call a point or a voxel an 'element'. The base network can be formulated as follows:

$$y = F_B(x), \tag{1}$$

where $F_B(\cdot)$ represents the base network, x is the input and y is the output. To meet the requirements of being stacked, x should be element-wise properties (including position, color, normal or other element features), and y should be one-hot encoded element-wise labels. For example, for a point cloud input with n points and position property only, x and y should be two matrixes with size of $n \times 3$ and $n \times l$ respectively where l denotes number of segmentation categories. In this way, the output of one base network can concatenate to another's input as additional context prior. Theoretically, any segmentation network that meets this restriction can be used as the base network. However, the base network should not be too complicated, because the overall complexity, model size and memory consumption will multiply as the base network is being stacked.

In our case, we use PointNet++ [QYSG17] as our base network. It handles point clouds and completely meets the requirements of being a basic network. It takes variable amount of point-wise properties as input and outputs point-wise semantic labels as segmentation result.

2.2. Stacking framework

We propose to stack the base network for multiple times for progressive 3D scene understanding, with the result of the former network passed to the latter one as additional contextual information.

The stacking framework is simple, as shown in Figure 1, the output of each base network is concatenated to the raw input and fed to the next base network. We call a stacked network with i base networks $F_i(\cdot)$, and obviously we have:

$$y_1 = F_1(x) = F_B(x), \tag{2}$$

which means stacking the base network for one time is the base network itself. And for $i > 1$, we can define our stacked network recursively:

$$y_i = F_i(x) = F_B(x \oplus F_{i-1}(x)), \tag{3}$$

where x is the raw input, $F_B(\cdot)$ represents the base network, $F_i(\cdot)$ represents the stacked network with i base networks, y_i is the output of the i-th base network, and $\oplus$ indicates element-wise concatenation. Suppose our final architecture is stacked by n base networks, we can get a hierarchical segmentation for the input 3D scene in the form of $\{y_1, y_2, ..., y_i, ..., y_n\}$, where y_n is the finest-grained segmentation result.

For the network training, a hierarchical supervision corresponding to the stacked architecture is required. Specifically, we need hierarchical ground truth labels in the form of $\{l_1, l_2, ..., l_i, ..., l_n\}$ for each scene in the training set, where l_i indicates the ground truth segmentation result of the i-th base network in the stacked architecture. The method of acquiring this kind of ground truth is described in Section 2.3, here we discuss the loss functions for training. We compute cross-entropy loss for each base network separately, and train them together with their weighted sum as the overall loss:

$$\mathcal{L}_i = H(y_i, l_i), \tag{4}$$

$$\mathcal{L}_{overall} = \sum_i \lambda_i \mathcal{L}_i, \tag{5}$$

where y_i and l_i denote the output and ground truth of the i-th base network respectively, $H(\cdot)$ is the cross-entropy function, $\mathcal{L}_i$ and λ_i represent the loss and weight of the i-th base network, and $\mathcal{L}_{overall}$ is the overall loss value.

In theory, the base network can be stacked any number of times as long as the corresponding hierarchical labels can be obtained. However in practice, most 3D scenes don't have that much objects to organize a very deep hierarchical structure. Moreover, the computational complexity and memory consumption of the entire model must be considered in the implementation. These facts restrict the maximum number of base networks that can be stacked. In our case, we stack the base network for 3 times, and their loss weights are set to 0.25, 0.25 and 0.5. The last network is more weighted because we want the finest-grained segmentation results to be better, while the former networks bias toward providing context prior rather than accurate coarse-grained segmentation results.

2.3. Hierarchical supervision

A hierarchical supervision is required for training the stacked network. In this paper, we generate hierarchical labels based on the contact relationships between objects. It is known that there is natural hierarchy in tree structure. Inspired by this, we generate hierarchical labels by merging objects based on a tree structure that describes the contact relationships between objects. The tree meets the following requirements: (1)Each node represents an object except the root node which is a virtual node introduced for ease of understanding; (2)An edge represents that the child object is attached to its parent object with a contact relationship; (3)The root node has no other child nodes except those with label 'ceiling', 'wall' or 'floor', because the room is composed of these three kind of objects, and all other objects are attached to them directly or indirectly. To construct such a tree structure meeting these requirements, we first build a connection graph. The graph represents a 3D scene, with each node represents an object and the weight of each edge represents the contact area between the bounding boxes of two objects. A root node is added to the graph and connected to the 'ceiling', 'wall' and 'floor' nodes with edges of infinite weight to ensure that the generated tree satisfies the above requirements. After this, a maximum spanning tree algorithm is employed to generate the tree structure.

With the tree that describes the contact relationships between objects, we can specify a merging depth d to obtain the segmentation result of the corresponding hierarchical level. For each node whose depth equals d, all its descendants are merged into it by assigning its label to these nodes. The labels of those nodes whose depth are smaller than d remain unchanged. The larger d is, the finer the segmentation result will be. In particular, setting d to 1 produces the coarsest-grained labels that segment the scene into 3 categories: 'floor', 'wall' and 'ceiling', which indicate objects on the ground, objects on the wall and objects on the ceiling respectively.

3. Experiments

We evaluate our method on the indoor fully reconstructed dataset: Stanford Large-Scale 3D Indoor Spaces(S3DIS) [ASZ*16]. This dataset contains 3D scans from Matterport scanners in 6 areas including 271 rooms with a total of 6020 square meters. The 6 areas are located at 3 buildings: Building 1 (Area 1, Area 3, Area 6), Building 2 (Area 2, Area 4), Building 3 (Area 5). This dataset is fully annotated for 12 semantic elements which pertain in the categories of structural building elements (ceiling, floor, wall, beam, column, window and door) and commonly found furniture (table, chair, sofa, bookcase and board). A clutter class exists as well for all other elements. We follow the official dataset split of training on 2 of the buildings and testing on the 3rd one.

3.1. The effectiveness of proposed method

We evaluate the effectiveness of the proposed method. For comparison, we try different strategies for stacking framework and hierarchical supervision. For the stacking framework, we stack the base network for 2 times as a comparison. For the hierarchical supervision, we generate the middle level labels by merging labels as a comparison. Note that the labels in S3DIS dose not support to form a 3-level hierarchical labels, so we generate 2-level hierarchical labels to compare with our strategy of merging objects.

Table 1: *Global accuracy of different stacking strategies*

Acc	Coarse grained	Medium grained	Fine grained
Base network (PointNet++)	-	-	70.41
Stack 2 times (merging labels)	80.27	-	71.66
Stack 2 times (merging objects)	84.61	-	72.80
Stack 3 times (merging objects)	84.80	71.02	75.03

The results are shown in Table 1 and Figure 2. It can be seen that the performance of base network can be enhanced with or without any additional strategies. Stacking for 3 times can get a better accuracy at the finest-grained level than stacking for 2 times. This is a matter of course, because stacking for 3 times has more network parameters to learn additional context prior. The merging objects strategy can also improve the result because the merging labels strategy may produce wrong coarse-grained labels. In terms of common sense, the finer-grained segmentation should be more difficult than the coarser-grained segmentation. However, the medium-grained accuracy of stacking for 3 times is worse than the fine-grained result. This is because: (1)the weight of loss for fine-grained output is larger than the weight of loss for medium-grained output, because we wish the finest-grained segmentation results to be better, while the former networks bias toward providing context prior rather than accurate coarse-grained segmentation results; (2)In our settings, only a few objects in the fine-grained segmentation are separated from the medium-grained segmentation, which means the gap between these two layers is not large while the fine-grained network has a more detailed prior than the medium-grain network.

3.2. Compared with other methods

In this section, we compare our method with PointNet [QSMG16], SEGCloud [TCA*17] and 3DCNN-DQN-RNN [LLZ*17].This section is for reference only, because: (1)Our goal is not beating state-of-the-art. We propose a strategy to improve a segmentation network, instead of a state-of-the-art framework; (2)The results of the mentioned methods are not provided by us, and some of them did not follow the official train/test split. But the results in Table 2 shows our approach is quite competitive with other methods. The results of PointNet and SEGCloud are from [TCA*17]. The PointNet is trained on 5 Areas and tested on the 6th Area. The SEGCloud, baseline and ours follow the official split of training on 2 of the buildings and testing on the 3rd one. The result of 3DCNN-DQN-RNN is from [LLZ*17], it removes the 'clutter' category and is trained on 70% rooms in every area while tested on the rest. The accuracy of 3DCNN-DQN-RNN on the category 'beam' is extremely high because of its train/test split. Almost all the 'beam' objects are located at Area 1 and Area 6 (both in Building 1), resulting in the 'beam' objects being either in the training set or in

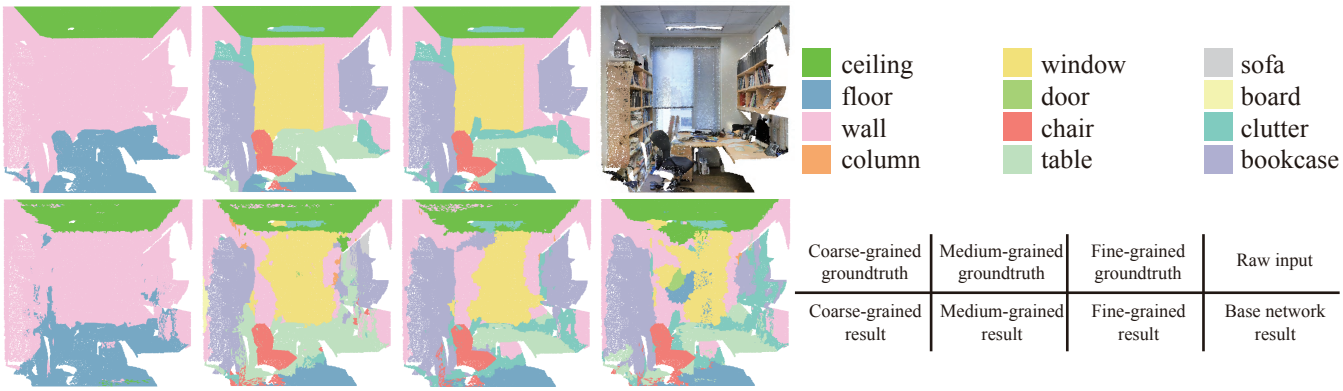

Figure 2: *Experimental results. We show a group of results including raw input, hierarchical groundtruths, hierarchical results produced by our stacked network and the result produced by the base network. The sequence of results is shown in the lower right corner.*

Table 2: *Comparison with other methods*

Method	ceiling	floor	wall	beam	column	window	door	chair	table	bookcase	sofa	board	clutter	mAcc	Acc
PointNet	88.80	**97.33**	69.80	0.05	3.92	46.26	10.76	52.61	58.93	40.28	5.85	26.38	33.22	48.98	-
SEGCloud	**90.06**	96.05	69.86	0.00	18.37	38.35	23.12	75.89	70.40	58.42	40.88	12.96	41.60	57.35	-
3DCNN-DQN-RNN	89.64	95.02	60.08	**78.55**	**89.36**	**75.29**	33.41	58.14	70.48	**84.97**	**76.98**	37.21	-	**70.76**	-
PointNet++ (baseline)	71.32	76.42	81.47	0.00	20.55	30.38	73.54	86.02	58.30	64.44	21.30	33.05	53.60	51.57	70.41
Stacked network	77.87	83.25	**84.39**	0.00	22.23	32.39	**74.89**	**87.70**	**75.88**	64.82	43.83	**46.56**	**56.29**	57.70	**75.03**

the testing set for other methods splitting by areas or buildings. Our method is fully superior than the baseline, which means our stacking framework can improve the base network.

4. Conclusions

In this paper, we propose a coarse-to-fine 3D scene understanding framework to understand 3D scene progressively with refined semantic classes by stacking off-the-shelf 3D scene segmentation networks. The former segmentation network understands 3D scenes at a coarser level and passes the result to the following network to provide effective contextual clues for the finer-grained understanding. We propose to generate hierarchical labels by applying a maximum spanning tree algorithm on the object connection graph, and train the stacked network with these hierarchical supervision end-to-end. Finally, using the trained model, we can not only obtain better segmentation accuracy at the finest-grained than directly using the segmentation network, but also obtain a hierarchical understanding results of the 3d scene as a bonus.

References

[ASZ*16] ARMENI I., SENER O., ZAMIR A. R., JIANG H., BRILAKIS I., FISCHER M., SAVARESE S.: 3D Semantic Parsing of Large-Scale Indoor Spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (jun 2016), IEEE, pp. 1534–1543. 3

[LLZ*17] LIU F., LI S., ZHANG L., ZHOU C., YE R., WANG Y., LU J.: 3DCNN-DQN-RNN: A Deep Reinforcement Learning Framework for Semantic Parsing of Large-Scale 3D Point Clouds. In *2017 IEEE International Conference on Computer Vision (ICCV)* (oct 2017), IEEE, pp. 5679–5688. 1, 3

[LZ17] LI Z., ZHANG J.: Pixel-level guided face editing with fully convolution networks. In *2017 IEEE International Conference on Multimedia and Expo (ICME)* (jul 2017), IEEE, pp. 307–312. 1

[NL13] NGUYEN A., LE B.: 3D point cloud segmentation: A survey. *IEEE Conference on Robotics, Automation and Mechatronics, RAM - Proceedings*, October 2015 (2013), 225–230. 1

[QLJ*17] QI X., LIAO R., JIA J., FIDLER S., URTASUN R.: 3D Graph Neural Networks for RGBD Semantic Segmentation. *Proceedings of the IEEE International Conference on Computer Vision 2017-Octob* (2017), 5209–5218. 1

[QSMG16] QI C. R., SU H., MO K., GUIBAS L. J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2016 Fourth International Conference on 3D Vision (3DV)* (dec 2016), 601–610. 1, 3

[QYSG17] QI C. R., YI L., SU H., GUIBAS L. J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems 30*, Guyon I., Luxburg U. V., Bengio S., Wallach H., Fergus R., Vishwanathan S., Garnett R., (Eds.). Curran Associates, Inc., 2017, pp. 5099–5108. 1, 2

[TCA*17] TCHAPMI L. P., CHOY C. B., ARMENI I., GWAK J., SAVARESE S.: SEGCloud: Semantic Segmentation of 3D Point Clouds. 1, 3

[ZSQ*17] ZHAO H., SHI J., QI X., WANG X., JIA J.: Pyramid Scene Parsing Network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (jul 2017), IEEE, pp. 6230–6239. 1

[ZYY*16] ZHANG Y., YING M. T. C., YANG L., AHUJA A. T., CHEN D. Z.: Coarse-to-Fine Stacked Fully Convolutional Nets for lymph node segmentation in ultrasound images. In *2016 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)* (dec 2016), IEEE, pp. 443–448. 1