Color guided convolutional network for point cloud semantic segmentation

Jing Yang¹,², Haozhe Li¹, Zhou Jiang³, Dong Zhang¹, Xiaoli Yue¹ and Shaoyi Du²

Abstract
Point cloud semantic segmentation based on deep learning methods is still a challenge due to the irregularity of structures and uncertainty of sampling. Color information often contains a lot of prior information, whereas the existing methods do not attach more importance to it. To deal with this problem, we propose a novel hard attention mechanism, named color-guided convolution. This convolution operator learns the correlation between geometric and color information by reordering the local points with color-indicated vectors. In addition, the global feature fusion is proposed to rectify features selected by the feature selecting unit. Experimental results and comparisons with recent methods demonstrate the superiority of our approach.

Keywords
Point cloud semantic segmentation, deep learning, color information, color guided convolution

Date received: 18 July 2021; accepted: 13 April 2022

Introduction
With the strong ability to reflect real scenes, three-dimensional (3D) data are getting more and more researchers’ attention. A point cloud is the main format of 3D data, and the semantic segmentation of the point cloud is the essential work for scene understanding, which is the key to the development of robots, autonomous driving, virtual reality, and remote sensing mapping. Inspired by the successes of deep learning methods in two-dimensional (2D) images and one-dimensional texts, many researchers have applied these techniques to analyze 3D point clouds.¹,² Unfortunately, it is difficult to use point clouds as direct input because they are intrinsically unstructured and disordered.³

The key of current deep learning methods for the semantic segmentation of point cloud is to construct ordered structures and then apply convolution operators to them. Recent research studies on construction methods can be mainly summarized as projection mapping, constructing graphs, and modeling local context.

Multi-view method⁴ and its variant⁵–⁷ project a global point cloud onto a regular structure, such as a 2D image. Two-dimensional convolution will be done on mapping results. Due to the self-sealing nature of the object’s spatial surface, it is likely to be many-to-one mapping, which

¹ School of Automation Science and Engineering, Faculty of Electronics and Information Engineering, Xi’an Jiaotong University, Xi’an, China
² Institute of Artificial Intelligence and Robotics, College of Artificial Intelligence, Xi’an Jiaotong University, Xi’an, China
³ School of Software Engineering, Faculty of Electronics and Information Engineering, Xi’an Jiaotong University, Xi’an, China

Corresponding author: Shaoyi Du, Institute of Artificial Intelligence and Robotics, College of Artificial Intelligence, Xi’an Jiaotong University, Xi’an 710049, China. Email: dushaoyi@xjtu.edu.cn
means prone to occlusion. Furthermore, this mapping rule is artificially selected, and distortion is inevitable. The method in Maxim et al.\textsuperscript{8} projects the local point cloud onto its tangent plane, then processes the projected image with two-dimensional convolution, and then adopts more reasonable projection rules but still relies on the estimation of the tangent plane. The voxel-based methods\textsuperscript{9–11} project point clouds onto 3D grids in Euclidean space. Sparse representation-based classification methods,\textsuperscript{12} for example, hash map, were also used to improve the retrieval performance. However, the convolution kernels of voxel-based methods and sparse representation-based classification methods are strictly limited to the grids, and the fine local structures are ignored by kernels. Although the above methods have achieved certain performance, information loss is inevitable, especially for local details, which play a decisive role in the understanding of complex scenes.

Graph convolution methods have also been applied to the point cloud segmentation task.\textsuperscript{13–15} These methods process unstructured data by constructing an adjacency relationship of point clouds, and the convolutions are performed on the graph adjacency relationship. The spectral CNN method\textsuperscript{15} enables weight sharing by parameterizing kernels in the spectral domain spanned by graph Laplacian eigenmodes. This spectral convolution usually requires expensive computations, and a spectral CNN model learned on one graph cannot be transferred to another graph that has a different Laplacian matrix. The authors in the literature studies\textsuperscript{13,16} define convolutions directly on a graph with local neighbors in a spatial domain, and the problem is formulated as a prediction on graph-structured data. However, their convolution weights are mainly generated according to the predefined local coordinate system, while neglecting the structure of the objects for semantic segmentation.

PointNet\textsuperscript{17} is the milestone to directly deal with the raw point cloud with the neural network. It inputs the whole point set into a shared multilayer perceptron, named MLP, for convolution. Although PointNet can handle unordered points, there are no local geometric contexts in PointNet, and sampling noise is not considered. The performance of PointNet is moderate. However, the authors of PointNet++\textsuperscript{18} integrate deep hierarchical feature learning on point sets with local context in the network. It is achieved by applying iterative farthest point sampling and ball query to group input points. KD-network\textsuperscript{19} first builds a KD-tree on input point clouds and then the hierarchical groupings are applied to model local dependencies in points. RSNet\textsuperscript{7} models in point clouds and designs the slice pooling layer to project features of unordered points onto an ordered sequence of feature vectors. Then, RNNs can be applied to them. PointCNN\textsuperscript{20} proposes to transform neighboring points to the canonical order and then applies convolution. KCNet\textsuperscript{21} improves the PointNet model by defining a set of learnable point set kernels for local neighboring points and presents a pooling method based on a nearest-neighbor graph. All these methods achieve promising results and show the ordered structures of local context are very important for point cloud semantic segmentation. However, there is still a gap between the performance of point cloud semantic segmentation and color segmentation of 2D images based on deep learning methods.

All the above methods focus on handling only geometric features on local point sets or parts of point clouds without using any color features. In human perception, sometimes color can be superior to geometric features when color can instantly make objects distinguished from the surrounding environment. Some researchers introduce the color information of point clouds into semantic segmentation. The authors in the literature\textsuperscript{3,21} use special context to reorganize points and take RGB as extra features. The work from Jiabo et al.\textsuperscript{22} proposes a semi-supervised prediction model, which exploits the improved unsupervised clustering algorithm to establish the fuzzy partition function, and then utilizes the neural network model to complete the future information prediction. The work from Jiachen et al.\textsuperscript{23} proposes a fully connected attitude detection network (FADN), which combines neural networks and traditional algorithms for 3D attitude angle estimation. FADN provides a whole process from the input of a single frame image in the industrial video stream to the output of the corresponding 3D attitude angle estimation. The convolutions of TangentConv\textsuperscript{8} are applied to them. However, the orientation of the tangent is estimated according to the local shape curvature, which is not stable because of curvature estimation in the local region. TangentConv evaluates convolutions on virtual tangent planes at every point and finds that adding RGB information can significantly improve the scores on Semantic3D. In addition, TangentConv takes additional depth, height, and normal information and combines them with color information, which means the combination of color and geometry is important for segmentation. The work from Verdoja et al.\textsuperscript{24} presents a novel fast method for 3D colored point cloud segmentation, and it starts with supervoxel partitioning of the cloud. Then, it leverages a novel metric exploiting both geometry and color to merge the supervoxels iteratively to obtain a 3D segmentation where the hierarchical structures of partitions are maintained. The work from Wang et al.\textsuperscript{25} produces predictions for points by similarity groups. The above four works simply regard original RGB as an input feature directly and require additional processing to obtain the improvement of performance.

To sum up, most of the existing works of point cloud semantic segmentation ignore the color characteristics of point clouds. Some works directly take the color information as the input and neglect the vital role of color in recombining geometric information. Usually, the change of color often reflects the change in the spatial characteristics of objects. Making full use of color information can
1. We propose a simple and efficient method for the reorganization of a local structure by color information.
2. We propose a novel network, named CGCN, to encode detailed geometric features where the color changes. Furthermore, the interaction of geometry and color is explored to make information fusion for semantic segmentation.
3. With the proposed CGCN, the color-spatial-fusion model for semantic segmentation is trained end to end and performs best among others with the same backbone network.

In the following parts of the article, details about the CGCN are presented in the second section. The third section reports all experimental results and the fourth section concludes.

**Color guided convolutional network**

The point clouds are unordered point sets with a format like $X = \{x_1, x_2, x_3, \ldots, x_n\}, x_i \in \mathbb{R}^d$. The candidate label set $L = \{l_1, l_2, \ldots, l_c\}$, and $c$ is the number of objects’ classes. For the semantic segmentation task, the aim is to assign every $x_i$ with $l_c$ in set $L$.

Considering the fact that geometry and color are not in the same distribution, we apply a network architecture to deal with geometric feature learning and color feature auxiliary encoding and use two cross paths to learn point coordinates and point color information, respectively. The network for coordinate information has more parameters to fit the complex geometric features of objects. The other network is relatively simple because the color information is an effective feature of semantic segmentation and too many parameters may bring overfitting. In fact, color information gives the relative position of objects from one or more categories. In local neighbors, indicator vectors represent this relationship. Furthermore, geometry and color information are merged for global fusion. Color information contributes to learning the distribution of points from different categories in geometric space, so as to get better semantic segmentation.

**Framework of CGCN**

In Figure 2, we give the presentation of our framework, which consists of two interlaced feature-encoding paths, the local encoding part and the color encoding part. The two encoding paths build several hierarchical feature abstraction levels. Each encoding level is composed of local fusion and color-indicated modules. The outputs of shared-MLP and color-indicated modules fuse together and are sent to the next level. After feature encoding, the extracted features are then fed to the feature interpolation module to obtain recovered features of a higher resolution. Then, the recovered features at each level are linked to the

---

| Figure 1. Reorganization of three directions in local neighborhoods. (a) Lighter color means a smaller cosine distance to the direction vector and (b) points are selected in each direction. | Yang et al. 3 |
encoded feature from the same resolution and sent to the next interpolation part. Afterward, the recovered geometry and color features for each point are followed by fully connected (FC) layers. Finally, the fused global features are followed by FC layers, and the prediction of each point is obtained.

The color-indicated module is the main function module in CGCN and is detailed in the next section. This module takes a $N \times (6 + d)$ matrix as input, which means $N$ points with 6-dim coordinates and a d-dim point feature. It outputs $N' \times (6 + d')$ matrix. After the $k$-nearest neighbor search, the inputs are first fed to the points selecting unit. The points, corresponding features, and related color vectors in three directions are selected and sent to color-indicated encoding layers, which are followed by the FC layer and pooling operation.

**Color-indicated module**

The key point for dealing with color information is how to use the color feature while there is a weak correlation or non-correlation between color consistency and geometric consistency. That means objects of the same or similar colors may be spatially independent. For example, a wooden door is the same color as a wooden table, but they are completely different in geometry and separated in space. We put the color-processing module in local neighborhoods, which can keep the continuity in space. In addition, color information gives the straightforward relative position of local points. Our encoding module is based on this relative position and digs local distributions of color and points. From this relative position, we get three vectors $d_1$, $d_2$, and $d_3$, which reorder the relationship of local points with the central point. The color-indicated module consists of two units, the points and the feature-selecting unit, and the color-indicated encoding unit. The former is to reorganize the points or features by color information, and the latter is to learn the reordered points and features.

**Points and feature selecting unit.** This part is the first step for color-encode in local neighborhoods of every sample point. The inputs of this unit are the unordered point clouds $p^m \in R^{N \times 6}$ and point features $f_p \in R^{N \times k \times d}$. The outputs are the features $F \in R^{N \times D}$, selected points $p_c \in R^{N \times 3 \times d}$, selected features $f_c \in R^{N \times 3 \times d}$, and low dimensional vectors $v = [v_s^1, v_s^2]$, which contain information on two color components $v_s^1, v_s^2 \in R^{N \times 3 \times d}$. $s = k/m$ is a hyper-parameter, denoting the number of points to select. $D$ is the dimension of the high-dimensional feature of the selected points. $c_1$ and $c_2$ are different color components, such as hue and saturation. $k$ is the number of local $k$ neighbors. $dc$ is the dimension of color information. $N'$ is obtained by farthest point sampling in Charles et al.\textsuperscript{18} We set $d = [d_1, d_2, d_3]$ as the directional vectors of local points. These vectors reflect the distribution of color differences in three directions of local neighborhoods. Radius neighbor search is better for ensuring robustness to different distributions of color information. We use the direction search unit to obtain the direction vector with abundant information in space and color. We set that $x$ is the point from point set $P, x \in R^{N \times 6}, f_i$ is the corresponding features from $F, f_j \in R^{N \times D}$, and $x_i$ is the $i$th neighborhood of $x$. For an arbitrary point $x \in R^{N \times 3}$, we get its relative position $\hat{y}^k = x_i - x$, $\hat{y}^k \in R^{N \times k \times 3}$, and neighborhoods $n^k = [y_i \in P^{N \times 3} | ||y|| \leq r (i = 1, \ldots, k); r$ is the known radius. In addition, $\hat{n}^k = [\hat{o}^{k}, \hat{c}^k]$ denotes the local $xyz$ and color in $\hat{n}^k$, respectively. From the color matrix $\hat{c}^k$, we get its standard deviation $\hat{s}^k$. Accordingly, $\hat{c}^k = [c_1, c_2, c_3]$ represents three color components, $c_1$, $c_2$, and $c_3$ in $\hat{c}^k$. The reorder operation for one component $c_1$ is calculated by the following equation

$$C_{ord} = \text{Order}([c_1, -\text{mean}(c_1)])^+$$  \hspace{1cm} (1)$$

where $c_1$ is the $i$th neighbor in $c_1$, Order$(\cdot)$ means sorting operation, and $t$ is a number, representing the power operation.

From ordered $C_{ord}$, we get the candidate directions from the first $k/4$ maximum values to the $k/4$ minimum values of the component $c_1$. 

---

*Figure 2.* Illustration of our CGCN architecture. CGCN: color guided convolutional network.
The first direction in $d^{1}_{i}/4$ for color component $c1$ is obtained by

$$d^{1}_{i} = \left( \min(||d^{1}_{i}/4||^2) \right)$$  \hspace{1cm} (2)

To reflect the whole special context, the second direction $d^{2}_{i}$ and the third direction $d^{3}_{i}$ are set opposite and orthogonal to $d^{1}_{i}$ as far as possible. $d^{1}_{i}, d^{2}_{i}, d^{3}_{i} \in \mathbb{R}^{N \times 1 \times 3}$. Thus, the local points are selected with more degrees of freedom. Furthermore, we use the angle between $\hat{v}^k$ and $d^{1}_{i}$ to represent one of the color descriptors $v^i$. That is $\theta^k = \hat{v}^k \cdot d^{1}_{i}$. The color descriptor vector for $c1$ is listed in the following equation

$$v^i = [s^k, \hat{v}^k, \theta^k]$$  \hspace{1cm} (3)

Vector $v^i$ depicts the local original color $\hat{v}^k$, standard deviation $s^k$, and the angle $\theta^k$ between the relative position vector $\hat{v}^k$ and direction $d^{1}_{i}$.

For points with different labels in a neighborhood, $d^{1}_{i}$ represents a relative position from one color to another. This direction vector guides the encoding module to associate the semantic feature with color change. $d^{2}_{i}$ gives the clue of inner points of identical objects related to the central point. The last direction ensures the preserving of information on the third dimension and implies where the boundary of the objects lies.

After getting the orientation vectors, the points $x_i \in P^n$ and corresponding features $f_i \in F$ in local neighborhoods are aggregated by cosine distance. That is to say, in local neighbors, the points $p^{c1}_{v} \in \mathbb{R}^{N \times s \times 6}$ with smaller cosine distance to $d^{1}_{i}$ and $f^{c1}_{v} \in \mathbb{R}^{N \times s \times D}$ corresponding features are selected, which is shown in the following equation

$$u^{c1}_{v} = \arg \left( \min \left[ \frac{\hat{v}^k \cdot d^{1}_{i}}{||\hat{v}^k|| \cdot ||d^{1}_{i}||} \right] \right), \ u = p, f, v$$  \hspace{1cm} (4)

The points in neighborhoods are encoded by the angles between relative position and color-indicated vector. With the same process, points, which are close to, and corresponding features are also selected and reordered as shown in Figure 3. For points with the same labels in a neighborhood, these vectors lead to the sample of points with various colors. The three directional vectors describe a distinct distribution of color, and they are rotation-invariant and robust to illumination effect.

**Color-indicated convolution.** As shown in Figure 4, the first layer in color-indicated encoding is the convolution of ordered vector $v^i$, input points $p^i \in \mathbb{R}^i$, and features $f^i \in \mathbb{R}^f$. The output of this part is the color-indicated feature $f^{c1}_v$. A kernel $g$ is defined on the selected points and features as follows

$$f^{c1}_v = g(v^i \times g^{1 \times 3})$$  \hspace{1cm} (5)

$$f^{c1}_f = g(f^i \times p^i \times g^{1 \times 3})$$  \hspace{1cm} (6)

Here, $v^i \times g^{1 \times 3} = \sum_{i=1}^{s} (g_1 v^i_{d1} + g_2 v^i_{d2} + g_3 v^i_{d3})$ means the convolution of color information in three directions, is the hue and saturation components of color, and $g(\cdot)$ is ReLU(BatchNorm($\cdot$)). The second layer is to deal with various components of color from the first point to $(s)$th points. The input is the results of the first layer, and the output is the color-indicated feature. To obtain a compact structure between points in the neighborhood, we use
1 × 3 convolution kernel for ordered points. In other words, the first point in a component of color and the first point in another component c2 join in convolution. The third layer consists of an FC layer and a dropout layer. Equation (7) means the concat of \( f^\text{c1}_F \) and \( f^\text{c1}_V \). The operations of two layers are defined in Equation (8). In fact, the previous layer gets repeated points from three directions. Therefore, we use the dropout layer for selected features \( f^\text{c}_F \), followed by the pooling layer, which is shown in Equation (9).

\[
\begin{align*}
\mathbf{f}^\text{c1}_F &= [\mathbf{f}^\text{c1}_F, \mathbf{f}^\text{c1}_V] \\
\mathbf{f}^\text{c}_F &= (\sum_{l=1}^{L} [\mathbf{f}^\text{c1}_F, \mathbf{f}^\text{c2}_F, \mathbf{f}^\text{c3}_F]) \times g^{1\times3} \times g^{1\times1} \\
\mathbf{f}^\text{D} &= \text{pooling}[\text{dropout}(\mathbf{f}^\text{c}_F)]
\end{align*}
\]

### Feature decoding and loss function

For better recovery of geometric and color features, a color feature interpolate module is used to decode points with distinct color features and point features. The interpolate feature interpolate module is used to decode points with the previous layer gets repeated points from three directions. The “3” in Equation (10) represents the sampling level of three layers

\[
f^j_l(x) = \frac{\sum_{i=1}^{3} w_i(x) f^j_l}{\sum_{i=1}^{3} w_i(x)}
\]

Here, \( C' \) is the number of output feature channels. In Equation (11), \( d \) represents the Euclidean distance, and the closer the distance is, the greater its weight is.

\[
d(x - x_i)^2
\]

Here, \( d \) represents the Euclidean distance, and the default value is 0.02. \( y_i \) represents the label of the predicted point.

### Experiments

#### Segmentation of benchmarks

Before being fed into CGCN, the point cloud needs to be processed. In real scenes, color information shows the design style or the type of material. However, the effect of the light may distort the real color to some extent. So, we hope preprocessing of color information can decrease noise as much as possible. To obtain such an effective and mutually independent color descriptor, we convert the color from RGB to HSV and just ignore the third component in HSV. From real-life experience, hue is the primary way to judge the difference between objects with various colors. Moreover, saturation is the secondary way. Thus, we choose these two components to complete preprocessing.

We evaluate the performance of CGCN on the Stanford 3D data set (S3DIS) and Semantic 3D. Two metrics, mean intersection over union (mIOU) and mean class recall (mRec), are used to measure the segmentation performance.

#### Training and inference details

Moreover, the number of neighbor points \( k \) is set to 32 for all data sets. The base learning rate was set to 0.001. The Adam solver is adopted to optimize the network with momentum set to 0.9. The above parameters in the part “Key parameter studies,” and , are the hue and saturation components of color, which means we translate color format from RGB to HSV and we only adopt the first two components of HSV. In the color-convolution block, there are two \( 1 \times 3 \) convolutional layers. The \( D_{out} \) in Figure 4 is the same as the number of output channels of the second \( 1 \times 3 \) layer.

#### Evaluation of S3DIS

For the S3DIS data set, the scenes are first divided into smaller cuboids using a sliding window of a fixed size on \( xy \) plane. A fixed number of points are sampled as inputs from the cuboid. In this article, the number of points is fixed as 4096 for the data set. Then, CGCN is applied to segment objects in the cuboids. During testing, the scene is similarly split into cuboids. We first run CGCN to get pointwise predictions for each cuboid and then merge predictions of cuboids in the same scene. We present the performance of CGCN on S3DIS. The training/testing split in Tchapmi et al.1 is used here to measure the generalization ability. Besides the overall mean IOU and mean accuracy, the IOU of each category is also listed. Some semantic results are shown in Figure 5. We list the test results in area 5 of S3DIS in Table 1 and our CGCN makes advances in most categories, such as floor (98.6%), table (79.7%), and window (56.2%). From the achieved advance, we notice that CGCN shows great superiority on objects with non-textual color changes, which is due to the color-induced encoding module. At the same time, we find that our method performs less well in the segmentation of door and column because there may mostly be no color change between the wall and these objects.

Table 2 presents the \( k \)-fold results on S3DIS. As shown in the results, our proposed CGCN can achieve the best performance with the semantic segmentation mean recall of 68.4%. Compared with the MLP- and RNN-based methods,3 we get a 1.9% improvement in mRec and 1.3% improvement in mIOU.
Evaluation of Semantic3D. The Semantic3D data set consists of 15 point clouds for training and 15 for testing. We only use the 3D coordinates and color information to train and test CGCN. The training/testing split in Hugues et al.\textsuperscript{29} is used here to measure the generalization ability.

Table 3 shows the segmentation results on Semantic3D, and our CGCN exerts advances in several categories, such as manmade terrain, natural terrain, and low vegetation. The color at the boundary of these objects changes, and the geometric structures of these categories mentioned above are relatively simple compared with other hardscapes. The segmentation result of buildings is not as good as expected. The prime reason is that different styles of buildings have different color distributions. In addition, the arrangement of windows brings trouble to get a stable color-indicated vector. We believe that the significant progress in combining geometry and color information depends on the efficient color-indicated encoding.

According to the experimental results, the color-indicated module in CGCN picks out some points with color changes and becomes a good supplement to the segmentation of geometric features, especially for those objects with distinct color differences and few textures. However, when an object is similar in color to its surroundings, such as a column, the color-indicated module learns unstable noise. In the S3DIS data set, some doors have the
same color as the walls, which brings a barrier to the segmentation of doors and walls. For outdoor scenes, compared with other categories of objects, the color texture of cars is more complex and diverse, and its segmentation accuracy is relatively low.

**Key parameter studies**

The proposed color-indicated encoding module is very important in our CGCN. In this section, we further deeply validate the effects of various parameters choices. In particular, several key parameters are considered: (1) the power in ordering operation, shown in Equation (1), and (2) the number of the selected points \( s = k/m \) in local neighborhoods, shown in Equation (4).

### Power in ordering operation

This hyper-parameter \( t \) is used to measure the change of various color components. It controls the offset to the mean \( t \) value of color components from \( k \) neighbors. In Table 4, we list the results of different values, 0, 1, and 2, with different ordering strategies of offsets. The results in Table 3 show that the power of \( t = 1 \) in ordering operation obtains the best performance for S3DIS data sets. The power \( t \) means the degree of deviation of local color so that a bigger \( t \) stretches this deviation and is weak at reflecting the original color feature.

### Number of selected points

The number of selected points means how many points to choose in three directions. If the value \( m \) in \( s = k/m \) is too small, some unnecessary points and extra features may be selected. On the contrary, if \( m \) is too big, there are few points or features that cannot reflect color differences in local neighborhoods, which makes the later convolution on features ineffective. In Table 4, the influence of differences \( m \) is presented. When \( t = 1 \) in CGCN, \( m = 4 \) achieves the best performance.

### Conclusion

In this article, we propose an efficient 3D segmentation network named CGCN. The key idea is to select points in local neighborhoods with color differences and guided vectors. More importantly, those vectors are only decided by color distribution and thus rotation-invariant. Experimental results show that CGCN achieves an effective combination of color and geometric information and can be applied to the semantic segmentation of large indoor and outdoor scenes.

**Author contribution**

Jing Yang and Haozhe Li contributed equally to this work.

**Declaration of conflicting interests**

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

**Funding**

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by the National Key Research and Development Program of China [Grant No. 2020AAA0108100], the National Natural Science Foundation of China [Grant Nos.
References

1. Tchapmi L, Choy C, Armeni I, et al. SEGCloud: semantic segmentation of 3D point clouds. In: Proceedings of the international conference on 3D vision (3DV), Qingdao, China, October 10-12, 2017, pp. 537–547.

2. Shelhamer E, Long J, and Darrell T. Fully convolutional networks for semantic segmentation. IEEE Trans Pattern Anal Mach Intell 2017; 39(4): 640–651.

3. Qiangui H, Weiyue W, and Ulrich N. Recurrent slice networks for 3D segmentation of point clouds. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, USA, June 18-21, 2018, pp. 2626–2635.

4. Hang S, Subhransu M, Evangelos K, et al. Multi-view convolutional neural networks for 3D shape recognition. In: Proceedings of the IEEE international conference on computer vision, Santiago, Chile, December 13, 2015, pp. 945–953.

5. Felix JL, Martin D, Patrik T, et al. Deep projective 3D semantic segmentation. In: Proceedings of the international conference on computer analysis of images and patterns, Ystad, Sweden, August 22-24, 2017, pp. 95–107.

6. Alexandre B, Bertrand LS, and Nicolas A. Unstructured point cloud semantic labeling using deep segmentation networks. In: Proceedings of the workshop on 3D object retrieval (3DOR), April 23-24, 2017, pp.17–24.

7. Xavier R, Jean-Emmanuel D, and François G. Classification of point cloud scenes with multiscale voxel deep network. arXiv preprint arXiv:1804.03583, 2018.

8. Maxim T, Jaesik P, Vladlen K, et al. Tangent convolutions for dense prediction in 3D. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, USA, June 18-21, 2018, pp. 3887–3896.

9. Wang W, Huang Q, You S, et al. Shape inpainting using 3D generative adversarial network and recurrent convolutional networks. In: Proceedings of the international conference on computer vision and pattern recognition, Honolulu, Hawaii, July 21-26, 2017, pp. 2298–2306.

10. Wu Z, Song S, Khosla A, et al. 3D shapenets: a deep representation for volumetric shapes. In: Proceedings of the international conference on computer vision and pattern recognition, Boston, Massachusetts, June 7-12, 2015, pp. 1912–1920.

11. Yizhak B, Michael L, and Anath F. 3dmv: three-dimensional point cloud classification in real-time using convolutional neural networks. IEEE Robot Automat Lett 2018; 3(4): 3145–3152.

12. Benjamin G, Martin E, and Laurens VDM. 3D semantic segmentation with submanifold sparse convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Salt Lake City, USA, June 18-22, 2018, pp. 9224–9232.

13. Wang Y, Sun Y, Liu Z, et al. Dynamic graph CNN for learning on point clouds. ACM Trans Graphics 2019; 38(5): 146.1–146.12.

14. Yi L, Su H, Guo X, et al. SyncSpecCNN: synchronized spectral CNN for 3D shape segmentation. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), Honolulu, HI, USA, July 21-26, 2017, pp. 6584–6592.

15. Chu W, Babak S, and Kaleem S. Local spectral graph convolution for point set feature learning. In: Proceedings of the European conference on computer vision (ECCV), Munich, Germany, September 8-14, 2018, pp. 52–66.

16. Wang LH, Yuchun H, Yaolin Z, et al. Graph attention convolution for point cloud semantic segmentation. In: Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, Los Angeles, USA, June 15-20, 2019, pp. 10288–10297.

17. Charles RQ, Hao S, Kaichun M, et al. PointNet: deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, Honolulu, Hawaii, July 21-26, 2017, pp. 652–660.

18. Charles RQ, Li Y, Hao S, et al. PointNet++: deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the international conference on neural information processing systems, Los Angeles, USA, December 4-9, 2017, pp. 5099–5108.

19. Roman K and Victor L. Escape from cells: deep Kd-networks for the recognition of 3D point cloud models. In: IEEE international conference on computer vision (ICCV), Venice, Italy, October 22-29, 2017, pp. 863–872.

20. Yangyan L, Rui B, Mingchao S, et al. Pointcnn: convolution on x-transformed points. In: Proceedings of the international conference on neural information processing systems, Montreal, Canada, December 3-8, 2018, pp. 820–830.

21. Federico M, Davide B, Jonathan M, et al. Geometric deep learning on graphs and manifolds using mixture model CNNs. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), Honolulu, HI, USA, July 21-26, 2017, pp. 5425–5434.

22. Jiabao W, Jiachen Y, Bin J, et al. Big data driven marine environment information forecasting: a time series prediction network. IEEE Trans Fuzzy Syst 2021; 29(1): 4–18.

23. Jiachen Y, Meng X, Bin J, et al. FADN: Fully connected attitude detection network based on industrial video. IEEE Trans Ind Inform 2021; 17(3): 2011–2020.

24. Verdoja F, Thomas D, and Sugimoto A. Fast 3D point cloud segmentation using supervoxels with geometry and color for 3D scene understanding. In: 2017 IEEE international conference on multimedia and Expo (ICME), Hong Kong, China, July 10-14, 2017, pp. 1285–1290.

25. Wang W, Yu R, Huang Q, et al. SGPN: similarity group proposal network for 3D point cloud instance segmentation. In: Proceedings of the IEEE conference on computer vision
26. Armeni I, Sener O, Zamir A, et al. 3D semantic parsing of large-scale indoor spaces. In: IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, June 26- July 1, 2016, pp. 1534–1543.

27. Hackel T, Savinov N, Ladicky L, et al. SEMANTIC3D.NET: a new large-scale point cloud classification benchmark. ISPRS Ann Photogramm Remote Sens Spatial Inf Sci 2017, pp. 91–98.

28. Chris Z, Wenjie L, and Raquel U. Efficient convolutions for real-time semantic segmentation of 3d point clouds. In: International conference on 3D vision (3DV), Verona, Italy, September 5-8, 2018, pp. 399–408.

29. Hugues T, François G, Jean-Emmanuel D, et al. Semantic classification of 3d point clouds with multiscale spherical neighbourhoods. In: International conference on 3D vision (3DV), Verona, Italy, September 5-8, 2018, pp. 390–398.