EasyChair Preprint

№ 843

Recognition of Point Sets Objects in Indoor Scenes

Ruizhen Gao, Xiaohui Li and Jingjun Zhang

March 19, 2019

Recognition of Point Sets Objects in Indoor Scenes

Ruizhen Gao, Xiaohui Li, Jingjun Zhang

Hebei University of Engineering, Handan, 056000, China
{ruizhenemail,lxh829}@163.com

Abstract. With the wide application of 3D intelligent sensing technologies in robotics and driverless driving field. Most researchers will transform point cloud data to regular 3D voxel grids, collections of images, depth maps, etc. which will inevitably lead to huge data processing problems. In this paper, we consider the problem of recognizing objects in indoor senses. We first use Euclidean distance clustering method to segment objects in indoor scenes. Then we use a deep learning network structure to directly extract features of the point cloud data to recognize the objects. Theoretically, this network structure shows strong performance. In experiment, there is an accuracy rate of 89.7% on the test set, this method is superior to current mainstream methods. Experiments show that the proposed network structure can accurately identify objects in indoor scenes and has strong robustness.

Keywords: Deep learning · 3D model recognition · Point cloud.

1 Introduction

Point cloud is a collection of points that express the spectral properties and spatial distribution of an object surface under the same Euclidean spatial reference system. Point cloud data include various information aspects such as time, position, distance, intensity, azimuth, angle, etc. It is an important part of the geometric data structure. In this paper, we use a network structure to directly process the point cloud data from objects in indoor scenes, and extract point cloud features in order to recognize the objects.

Compared with other methods, most of the objects in point cloud scene model in this paper have no repeated occlusion. Euclidean distance clustering segmentation methods can be used to segment objects in complex scenes. The samples n points require as much original feature information as possible, so Monte Carlo method is used. Using a deep learning network that directly process point cloud can greatly reduce the amount of data calculation. Point cloud does not introduce quantization artifacts, which can better maintain the natural invariance of data. Experiments show that the combination of clustering and deep learning networks can identify most objects in indoor scenes with high accuracy and robustness.

The network structure improved in this paper is a systematic method, which directly uses n sample points from the point cloud data as the input to the network structure. Normal calculations and extracted local or global features can be added to other dimensions to classify and recognize the input point cloud data. In the basic setting each point of the input is processed identically and independently, and is represented by just its three coordinates (x, y, z).

In this paper, the indoor scene is visualized in Fig 1. Point cloud image of indoor scene is visualized in Fig 2. Point cloud image after segmentation by Euclidean distance clustering is visualized in Fig 3.

The key works of this paper are as follows:

•Use the Euclidean distance clustering segmentation method to divide multiple objects in indoor scenes into clusters and perform unified data processing. The same and independent processing of data by Monte Carlos sampling method, with zero mean and normalization.

•Use a deep learning network architecture that directly consumes irregular point sets to complete the recognition task.

•Provide an analysis for the accuracy of object recognition in the indoor scene using the improved network method and to evaluate the robustness of the network method.

Fig. 1. The indoor scene in this paper.

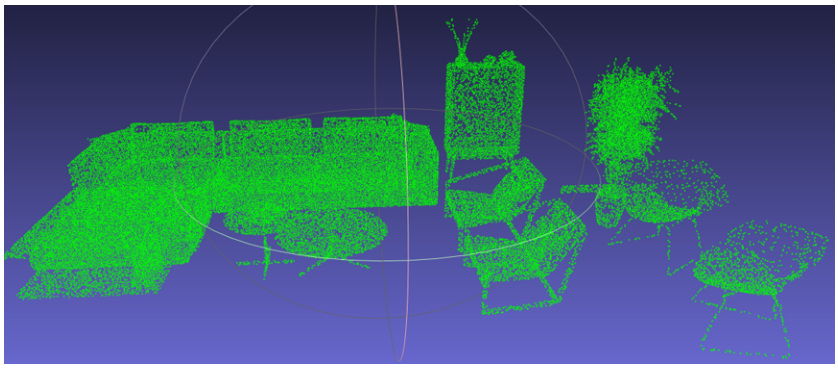

Fig. 2. Point cloud image of indoor scene.

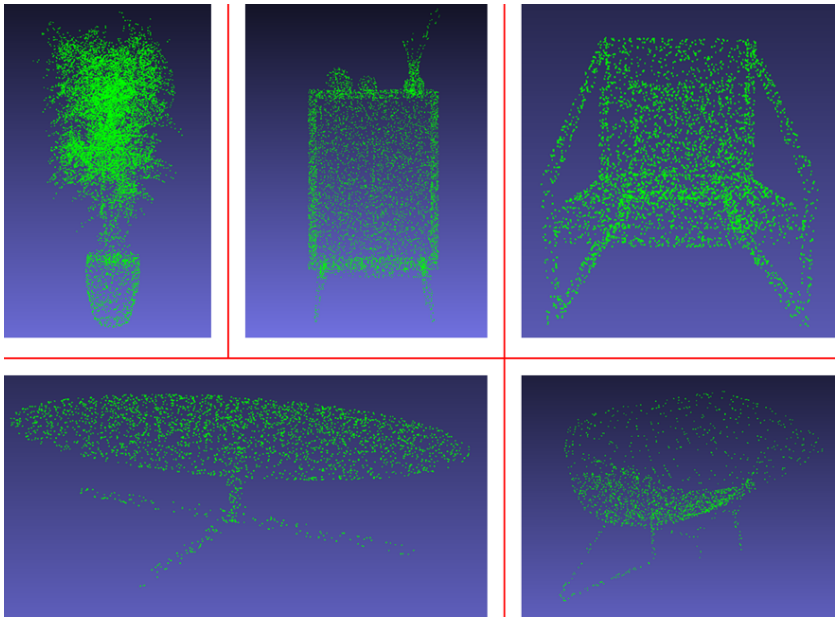

Fig. 3. Point cloud image after segmentation by Euclidean distance clustering.

2 Related Work

3D data has multiple popular representations, leading to various approaches for learning. There are three main methods for 3D object recognition, which are based on 3D voxel grid, collections of images and point cloud data. In addition to these methods, there are some other methods such as spectral convolutional neural network (CNN), feature-based deep neural network (DNN), etc.

Methods based on collections of images data, transform 3D multiple images into a 2D convolutional network structure based on two-dimensional images, and then add three-dimensional information to the two-dimensional network structure. [13] attempt to render 3D point clouds or shapes as 2D images, combining multiple view information from 3D shapes into a single and compact shape descriptor by providing a novel CNN architecture. [20] uses a multi-view approach; more than 500,000 models of more than 55 common categories in Shape-Net Core55 are used to train and evaluate multiple algorithms. These methods are evaluated on several standard information retrieval performance indicators.

Methods based on the 3D voxel grid data is to perform voxelization on the data, and then use 3D convolution network for feature extraction and analysis. [18,15,10,16,13] are some 3D convolutional neural network applied to voxel shapes. However, because data sparsity and 3D convolution require huge computational costs, the data format of

3D voxel grid is limited by its resolution. FPNN [8] and Vote3D [17] propose different methods for solving sparsity problems. However, their operations are still sparse and therefore still a challenge for dealing with very large point cloud data.

Methods based on the point cloud data have two approaches: one way is transforming the point cloud data to a 3D data form of the voxel grid (described above), and another way is directly processing the point cloud data. The second approach has been more accurate and enjoyed a more rapid development trend than other methods in the last three years. [12] believe that converting 3D data into voxel grid, depth map or multi-view will lead to unnecessary huge problems of data. Therefore, they propose new network architecture PointNet[12] and PointNet++[14] that directly consumes point cloud. [6] proposes a new deep learning architecture K-d-network that is designed for 3D model recognition tasks and works with unstructured point clouds. [21] mainly characterizes the permutation invariant function, which provides a series of functions that can be run on the set. This function has applicability in a variety of locations. The best effect is that researchers at Shandong University proposed PointCNN [7] to use the convolution operator in convolutional networks. PointCNN uses χ transform to weight the input features associated with points, which works very well in classification and segmentation scene.

For spectral CNN [2,9,11], these measures are currently limited to meshes such as organic objects. Feature-based DNN [3,4] converts 3D data into vectors by extracting traditional shape features, then uses a fully connected network to classify shapes on point cloud data.

3 Problem statement

The first problem: how to separate point cloud data from multiple objects. In the indoor scene created in this paper, there are multiple objects, but they are all in the same file. Split multiple objects into individual objects based on distance, texture, color, and other information. The point cloud data is saved to a separate file. The point cloud data of a single object is identified using a well-trained deep learning network.

The second problem: how to use deep learning on point sets. For the recognition of point cloud objects in an indoor scene, the Euclidean distance is used to pre-segment the point cloud from the scene point cloud. Sampling point cloud data after segmentation using Monte Carlo method. The deep learning network used is to directly consume the unordered point set as input. The point cloud is represented as a set of 3D points $\{P_i|i = 1, \cdots, n\}$, where each point$\mathrm{P_i}$ is its (x, y, z) coordinate plus additional feature channels such as color, normal. For the sake of simplicity and clarity, this paper uses only the three coordinates (x, y, z) as input to the network.

4 Euclidean distance clustering segmentation and data preprocessing

Euclidean distance clustering segmentation and data preprocessing are divided into two parts. First, the working principle of Euclidean distance clustering segmentation

algorithm is introduced (Sec 4.1), The method has a better effect in an indoor scene with less overlap. Second, the same and independent processing of data by Monte Carlos sampling method, zero mean and normalization is introduced (Sec 4.2).

4.1 Euclidean distance clustering segmentation

The Euclidean distance clustering algorithm is essentially done by distinguishing the distance between neighborhoods. Point cloud data can provide higher dimensional information. More information can be obtained by extraction. In this paper, the Euclidean distance between the neighborhoods is used as a criterion. The Euclidean distance in n-dimensional space is:

$$\begin{aligned} \rho &= \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)^2} \\ &= \sqrt{\Sigma(x_i - y_i)^2}, \end{aligned} \tag{1}$$

Since it is a three-dimensional space in this paper, the Euclidean distance is:

$$\rho = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2} \tag{2}$$

The specific implementation method is: Calculate the distance d_{ij} from each point to the remaining points for all data points, calculate the density using the formula: $\rho_i = \sum \beta(d_{ij} - dc)$, where $\beta(x)$ is a sign function, $\beta(x) = 1$ when $x < 0$, and $\beta(x) = 0$ when $x >= 0$. Find the maximum density point of ρ_i, whose Euclidean distance is δ_i, and the data points with larger ρ_i and δ_i values as the cluster center point. The appropriate threshold r is selected according to the data of the clustering segmentation scene, and the most suitable threshold r is selected by setting different threshold values, 1) find a cluster center point p_1 in space, through k-dTree (a recursive proximity search strategy) find the n points closest to it, judge the distance from the n points to p_1, put the points $p_2, p_3...$ whose distance is less than the threshold r into class A; 2) find a dot p_2 in $A \backslash p_1$,repeat 1); 3), then $A \backslash p_1 p_2$ find a dot, repeat 1), find $p_7, p_8...$ all into A; 4), when A no longer has a new dot to join, the search is completed. The result of the segmentation is shown in Fig 3.

4.2 Data processing

Since the number of points of the point cloud data of the divided single object is different, the point cloud data needs to be sampled into a certain number of points (n). The n points are zero mean and normalized. The data is converted into a uniform format for input into the network. It is very difficult to find accurate answers for sampling these points. In this paper, we use down sampling, uniform sampling, and Monte Carlo sampling to compare the results of sampling the same object and the time required to complete the sampling. In Fig 4 can be clearly seen that uniform sampling and Monte Carlo sampling can better maintain its external features. However, uniform sampling takes more time and cost in the case of a larger amount of data. Considering comprehensively, this paper uses Monte Carlo approximation to sample.

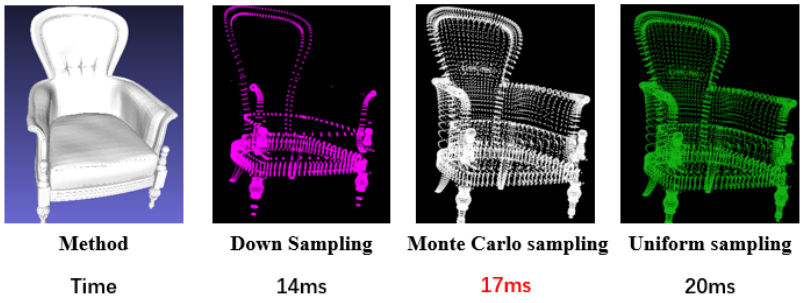

Fig. 4. Comparison of three sampling methods.

Monte Carlo thought is easy to sample and calculate similar results on point cloud data. With the increase of sampling points, the closer the results are to the correct results, the better the sampling points can be matched with the original point distribution. The theory, proof, and algorithm are implemented as follows:

Assuming the given an approximation of function $h(x)$, the integral needs to be calculated is:

$$\int_a^b h(x)dx = s, \tag{3}$$

for the mathematical derivation can't directly find the solution, but avoid enumerating all the x values on the interval (a, b), decompose $h(x)$ into a function $f(x)$ and a definition in (a, b) the product of the probability density function p, the entire integral can be equivalently written as:

$$s = \int_a^b f(x)p(x)dx = E_p[f(x)], \tag{4}$$

where p is a probability distribution about the random variable x, E_p is the expectation of p. In this way, the original integral is equivalent to the mean of $f(x)$ over the distribution of p. At this time, if we collect a large number of sample points $x^{(1)}, x^{(2)}, \cdots, x^{(n)}$ a from the distribution p to approximate s, we obtain an empirical average:

$$\hat{s_n} = \frac{1}{n}\sum_{i=1}^{n} f(x^{(i)}), \tag{5}$$

the mean is approximated by the collected samples:

$$s = \int_a^b h(x)dx = E_p[f(x)] \approx \frac{1}{n}\sum_{i=1}^{n} f(x^{(i)}) \tag{6}$$

The following properties indicate the rationality of this approximation. It is also easy to observe that $\hat{s}$ this estimate is unbiased,

$$E[\hat{s_n}] = \frac{1}{n}\sum_{i=1}^{n} E[f(x^{(i)})] = \frac{1}{n}\sum_{i=1}^{n} s = s, \tag{7}$$

according to the Law of Large Numbers, if the sample $x^{(i)}$ is independently distributed, its mean must converge to the expected value, that is:

$$\lim_{n\to\infty} \hat{s_n} = s, \tag{8}$$

it is only necessary to satisfy the variance of each individual item $Var[f(x^{(i)})]$ bounded. In detail, we consider that the variance of $\hat{s_n}$ when n increases, as long as $Var[f(x^{(i)})] < \infty$ is satisfied, the variance $Var[\hat{s_n}]$ will decrease and converge to 0:

$$Var[\hat{s_n}] = \frac{1}{n^2}\sum_{i=1}^{n} Var[f(x)] = \frac{Var[f(x)]}{n}. \tag{9}$$

Zero mean and normalization (which can get better experimental performance) is a necessary step before training the neural network. Zero mean and regularization can accelerate the convergence of weight parameters during training. Zero mean refers to the use of data minus the mean of the data. Assuming the data sample is x_i where $i = 1, 2, , n$ and the mean of the data samples is u, then the data zero mean is calculated using the following formula:

$$x_i^{'} = (x_i) - \mu, i = 1, 2, 3, ..., n. \tag{10}$$

To scale the original data, we use Min-Max scaling to convert the original data linearization to $[1, 1]$, as follows:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}, \tag{11}$$

where X_{norm} is the normalized data, X is the original data, X_{max} and X_{min} are the maximum and minimum values of the original data set, respectively.

5 Deep learning on point sets

Deep learning on point sets are divided into two parts. First, Sec 5.1 introduces two main problems, solutions, proofs in the process of deep learning processing point sets. Second, Sec 5.2 introduces the improved network structure for identifying objects.

5.1 Problems of point sets in $\mathbb{R}^n$

The input point cloud data used in paper is a subset of points from the Euclidean space. The following are two problems encountered in the processing of point cloud data and the corresponding solutions.

•Unordered. Unlike pixel arrays in images or voxel arrays in volumetric grids, point cloud is a set of points without specific order. In other words, when writing a set of point cloud arrays in different orders, whether it is classification, semantic segmentation, and scene segmentation, the same result must be output. In 2D images, the pixel position of the point is relatively invariant, and there is no such problem of unordered points. However, for point cloud data, there are $N!$ types of input arrangement of points, it must be processed.

There are two ways to solve the problem of the unordered of input of point cloud data; 1)sort input into a canonical order; 2)use a simple symmetric function to aggregate the information for each point. Although the input point cloud data uses a regular order like a simple solution, there is actually no stable ordering in high dimensional space. This can be easily shown by contradiction. If there is such a sorting strategy, it defines a bijection map between the high dimensional space and the one-dimensional real line. It is not difficult to see that the sequence is required to be stable. The point perturbation is equivalent to requiring the map to maintain spatial proximity when the dimension is reduced. This is a task that cannot be realized under normal circumstances. Therefore, sorting does not completely solve the sorting problem. This method refers to the PointNet classification network [12]. The solution in this paper is to approximate the general function defined on the point set by applying a symmetric function:

$$f(\{x_1, \cdots, x_n\}) = \gamma \circ g((h(x_1), \cdots, (h(x_n))) \tag{12}$$

where $f\ \ 2^{\mathbb{R}} \rightarrow \mathbb{R}$, h represents the feature extraction layer, g represents the symmetric method using max-pooling, γ represents higher dimensional feature extraction. The basic module used in this article is very simple: approximate h by a multi-layer perceptron network, approximate f by a combination of a single variable function and a max-pooling function. Through the collection of h, network can learn some f to capture the different properties of the collection. That is, the final high-dimensional feature is the corresponding maximum eigenvalue among the n points selected in each dimension. g can be used to solve the problem of disorder of point cloud data.

Theorem 1. *Suppose f is a continuous set function about Hausdorff distance d_H, $\forall \varepsilon > 0$, $\exists$ a continuous function h and asymmetric function $g(x_i, ..., x_n = \gamma \circ Max)$, such that for any $W \epsilon Q$,*

$$|f(W) - \gamma(\max_{x_i \epsilon W}\{h(x_i)\})| < \varepsilon, \tag{13}$$

where $x_1, ..., x_n$ is a complete list of any ordered elements in W, γ is a continuous function. Max is a vector max operator that takes n vectors as input and returns a new vector of the elements maximizing value of function g. Using the max-pooling layer, after convolving operations on n points, the corresponding maximum value can be obtained for each dimension. The problem of input disorder of point cloud data can be solved.

Proof. let $Q = \{W : W \subseteq [0,1]^m and |W| = n\}$, $f : Q \rightarrow \mathbb{R}$ is a continuous set function on Q about Hausdorff distance d_H, $\forall \varepsilon > 0$, $\exists \delta > 0$, for any $W, W^{'} \epsilon Q$, if $d_H(W, W^{'}) < \delta$, then $|f(W) - f(W^{'})| < \varepsilon$. This theorem says that f can be arbitrarily approximated by network given enough neurons at the max-pooling layer.

•Invariance under transformations. As a geometric object, the learning representation of a point set should be invariant to some transformations. For a three-dimensional data, when we perform operations such as rotation and translation, we should ensure that the results are unchanged whether it is classified or segmented.

For the invariance of point cloud data after rotation or translation, a natural solution is to align all input sets to the canonical space before feature extraction. Jaderberg et al. [5] introduces the idea of spatial transformer to align 2D images by sampling and interpolation, through a specially customized layer implemented on the GPU. This paper uses the adjustment network proposed by Charles R. Qi et al. [12] to solve this problem, and the effectiveness of this solution has been proved in the paper. By adjusting the network prediction affine transformation matrix and applying the transformation directly to the coordinates of the input points, the idea can be further extended to the consistency of the feature space. However, the transformation matrix in the feature space has a higher dimension than the spatial transformation matrix, which greatly increases the difficulty of optimization. Therefore, it is necessary to add a regularization term to the soft-max training loss to approximate the feature transformation matrix to the orthogonal matrix:

$$L_{reg} = \|I - AA^T\|_F^2 \tag{14}$$

where A is the characteristic alignment matrix of the adjustment network. This orthogonal transformation does not lose the input information.

By adding the regularization term, the optimization solution becomes more stable. The transformation matrix is approximated to the orthogonal matrix, which can greatly reduce the parameters.

The schematic diagram of adjusting the network structure is shown in Fig 5.

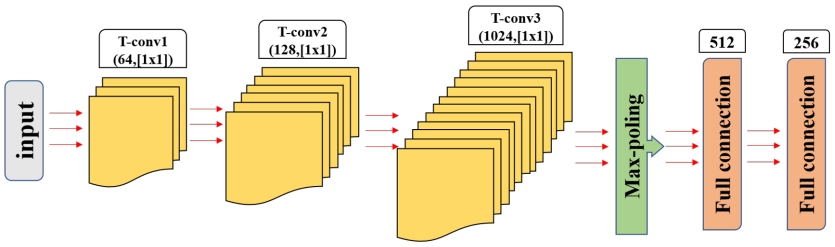

Fig. 5. Adjust network structure diagram.

5.2 Network architecture

Because PointNet [12]s classification and segmentation network has achieved good results, and is ability to extract features is very strong, it motivates our recognition requirements for point cloud objects in indoor scenes. This article draws on the classification network of PointNet. The adjustment of the structure of this network in this

paper makes the network further enhance the feature extraction ability. The experimental results show that the improved network structure can not only improve the accuracy of the network to identify a single point cloud model, but also improve the accuracy of the same test data set. The improved network structure to perform point cloud data recognition tasks for objects in door scenes, the network structure is shown in Fig 6. By inputting the three-dimensional coordinates of the n points of the point cloud data. The network is adjusted to avoid the invariance of rotation or translation. The feature is extracted by the combination of two multi-layer perceptron and the adjustment network. In addition, a multi-layer perceptron is further extracted from the feature. The global feature of the output is obtained through the max-pooling layer. The output is the score corresponding to each category of the object. Through this process, the point cloud object recognition in the indoor scene is completed.

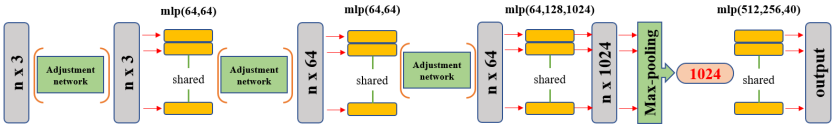

Fig. 6. Complete network structure diagram.

6 Experiment

Experiments are divided into two parts. First, Sec 6.1 provide detailed training process. Second, Sec 6.2 analyze the experimental results and test the robustness of the network.

6.1 Training process

This article uses the ModelNet40 [19] shape classification benchmark established by Stanford University to train and test deep learning networks. This has a total of 40 categories (each with a corresponding number corresponding to it) CAD model (12311 total). The total CAD models are divided into training and test sets. We randomly select 8192 model files as the training set. We then divide these models into 4 files with, 64 group per file and 32 models per group. The remaining 2048 model files are used as test sets. They were written in a file in the same way. During training we augment the point cloud on-the-fly by randomly rotating the object along the up-axis and jitter the position of each points by a Gaussian noise with zero mean and 0.02 standard deviation.

The device used to train the deep learning network is an Inter Core $i7-5960X$ processor, $16G$ ROM, GPU $M4000$, and running Ubuntu 18.04 and TensorFlow $1.7.0$ [1] computers it costs 17 hours on this platform to complete the entire training process after tens of thousands of iterations. The entire training process is recorded as shown in Fig 7.

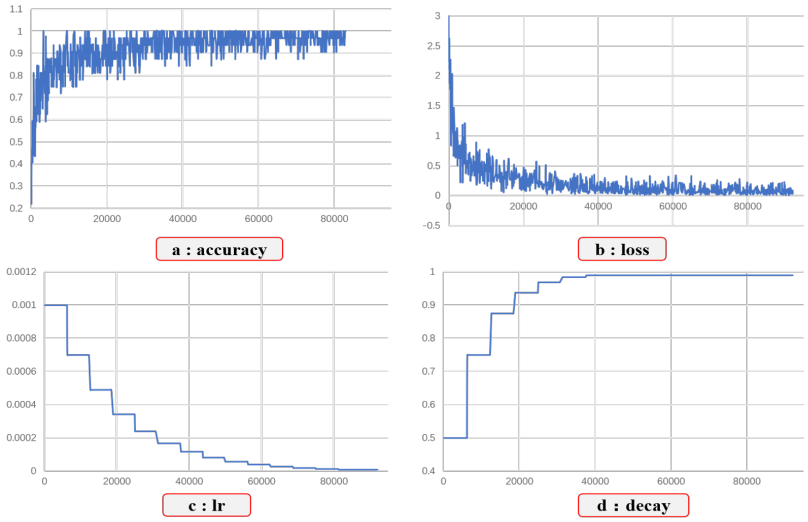

Fig. 7. The entire training process.

Because different results are obtained after the initial parameter setting of the same network, the starting and ending values of some important parameters are recorded in Table 1.

Table 1. The initial and end values of some parameters during the training process.

	Initial value	End value
accuracy	0	0.988
Cross entropy loss	1.9408	0.024
Learning rate	0.001	0.0000133
Decay rate	0.5	0.982

6.2 Results and analysis

After the training is completed, this paper will test the trained deep learning network using the test set data. The average loss is 0.5014, the average accuracy is 0.897. In Table 2, this paper compares this model with previous work. The model in this paper has better performance in 3D input and can easily parallelized uses similar CPU. Because it only contains full connected layers and max-pooling layers.

The task of this paper is to identify point cloud objects in indoor scenes. Table 3 below is an analysis of the accuracy of recognition objects that may appear in indoor scenes.

Table 2. Comparisons of classification accuracy (%) on ModelNet40.

Method	Input	ModelNet40
FPNN [8]	Volume	68.20%
3DShapeNets [19]	Volume	77.30%
VoxNet [10]	Volume	83.00%
Subvolume [13]	Volume	86.00%
PointNet(vanilla) [12]	Point Cloud	87.20%
PointNet [12]	Point Cloud	89.20%
This paper	Point Cloud	**89.70%**

Table 3. Results on the test set.

Bookshelf	Lamp	Chair	Cup	Plant	Sofa
0.91	0.95	0.98	0.70	0.80	0.97
Wardrobe	Guitar	Keyboard	Bottle	Bowl	Curtain
0.55	0.94	0.85	0.94	0.99	0.90
Door	Laptop	Person	Piano	Desk	Bed
0.85	0.99	0.95	0.87	0.99	0.99

It can be seen from Table 3 that some objects with more obvious features have a higher recognition rate and a lower accuracy for less obvious object features. Overall, it is still possible to recognition objects in the indoor scenes. Some example of identification is shown in Fig 8.

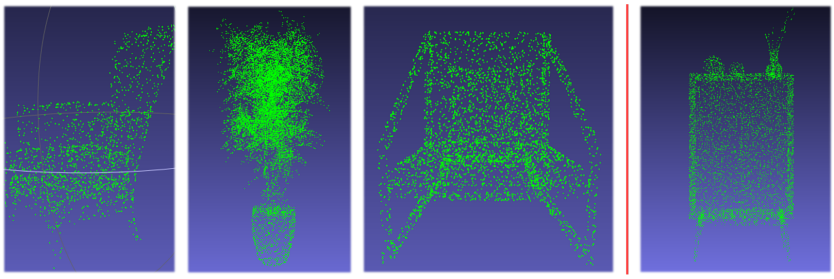

Fig. 8. Give examples of identifying the right objects and identifying the wrong objects. The correctly identified object is the three objects on the left as shown. The object on the right as shown is wrong recognition, the correct label should be the wardrobe, but the result of the network is the table. The main reason is that there are similar external features. This is inevitable.

This paper test the robustness of the network by comparing the number of points in the network to the accuracy of the network on the test set. The test results are shown in Fig 9.

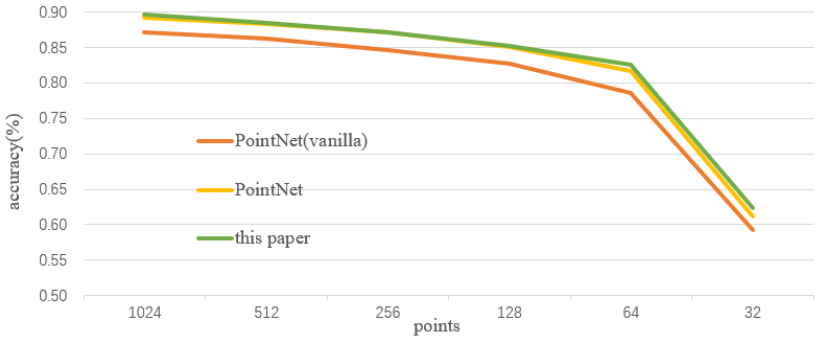

Fig. 9. Robustness Test. It can be observed that as the number of points in the input network decreases, the accuracy of the network on the test set becomes lower and lower. As long as the number of input points is greater than or equal to 64 points, the accuracy of the test set can be maintained above 80%. It can be seen that the robustness of the network is still very good.

7 Conclusion

This paper proposes a new approach to recognize point sets objects in indoor scenes by clustering and segmenting point cloud data in indoor scenes using on the Euclidean distance clustering segmentation algorithm. It effectively solves the problem of clustering and segmentation of multiple objects in complex scenes. Using a deep learning network that directly processes point cloud greatly reduces the amount of data calculation. Point cloud does not introduce quantization artifacts, which can better maintain the natural invariance of data. Experiments show that the combination of clustering and deep learning networks can identify most objects in indoor scenes with high accuracy and robustness. Since the network structure used in the paper only extracts the global features and does not make good use of the local features, in the subsequent work, we hope to improve the structure of the network and make good use of the local features of the point cloud data. Further improve the accuracy of single target classification and recognition in indoor scenes.

Acknowledgments. This research was supported by Natural Science Foundation of Hebei Province of China (No. F2016402106, No. F2017402182) and Science and technology research projects of Colleges and Universities in Hebei, China (No. ZD2018207).

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: a system for large-scale machine learning. In: OSDI. pp. 265–283 (2016)
2. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. arXiv preprint arXiv:1312.6203 (2013)
3. Fang, Y., Xie, J., Dai, G., Wang, M., Zhu, F., Xu, T., Wong, E.: 3d deep shape descriptor. In: Proc. CVPR. pp. 2319–2328 (2015)

4. Guo, K., Zou, D., Chen, X.: 3d mesh labeling via deep convolutional neural networks. ACM Transactions on Graphics (TOG) (1), 3 (2015)
5. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. In: Advances in neural information processing systems. pp. 2017–2025 (2015)
6. Klokov, R., Lempitsky, V.: Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In: ICCV. pp. 863–872. IEEE (2017)
7. Li, Y., Bu, R., Sun, M., Chen, B.: Pointcnn. arXiv preprint arXiv:1801.07791 (2018)
8. Li, Y., Pirk, S., Su, H., Qi, C.R., Guibas, L.J.: Fpnn: Field probing neural networks for 3d data. In: Advances in Neural Information Processing Systems. pp. 307–315 (2016)
9. Masci, J., Boscaini, D., Bronstein, M., Vandergheynst, P.: Geodesic convolutional neural networks on riemannian manifolds. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 37–45 (2015)
10. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on. pp. 922–928. IEEE (2015)
11. Monti, F., Boscaini, D., Masci, J., Rodola, E., Svoboda, J., Bronstein, M.M.: Geometric deep learning on graphs and manifolds using mixture model cnns. In: Proc. CVPR. p. 3. No. 2 (2017)
12. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proc. CVPR (2), 4 (2017)
13. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view cnns for object classification on 3d data. In: Proc. CVPR. pp. 5648–5656 (2016)
14. 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 Natural Information Processing Systems. pp. 5099–5108 (2017)
15. Riegler, G., Ulusoy, A.O., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: Proc. CVPR (2017)
16. Shao, T., Yang, Y., Weng, Y., Hou, Q., Zhou, K.: H-cnn: Spatial hashing based cnn for 3d shape analysis. arXiv preprint arXiv:1803.11385 (2018)
17. Wang, D.Z., Posner, I.: Voting for voting in online point cloud object detection. In: Robotics: Science and Systems. No. 3 (2015)
18. Wang, P.S., Liu, Y., Guo, Y.X., Sun, C.Y., Tong, X.: O-cnn: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics (TOG) (4), 72 (2017)
19. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proc. CVPR. pp. 1912–1920 (2015)
20. Yi, L., Su, H., Guo, X., Guibas, L.J.: Syncspeccnn: Synchronized spectral cnn for 3d shape segmentation. In: Proc. CVPR. pp. 6584–6592 (2017)
21. Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R., Smola, A.J.: Deep sets. In: Advances in Neural Information Processing Systems. pp. 3391–3401 (2017)