The 33rd Annual Conference of the Japanese Society for Artificial Intelligence, 2019

Local Feature Fitting Learning Network for Point Cloud Classification

Lu SUN [*1] Yoshitsugu MANABE [*1]

[*1] CHIBA University

We propose a deep learning network framework to solve the classification task of three-dimensional point clouds. According to different functions, the network can be divided into the resampling block, transform block, local feature fitting block, and classification block. Unlike other classification methods based on point cloud, we try to fit local point cloud and use the fitting function as a local feature to enter the classification layer. Though simple, Local feature fitting learning network (LFFLN) is highly efficient and effective. It achieves excellent performance in the ModelNet40 without any tricks.

1. Introduction

In recent years, deep learning tools have made great achievements in image-based works. However, it is difficult to apply the image-based network framework to three-dimensional models directly. Therefore, the accuracy of image classification is more than 99%, while the accuracy of the three-dimensional model classification is mostly hovering at 85%. The main problem is that point clouds cannot be convoluted directly. The main contribution of this study is to simulate the convolution operation with the local fitting of point clouds and to find the local characteristics of point clouds.

2. Related Work

2.1 Non-point Cloud Method

To use convolution in three-dimensional models, some researchers use voxels to represent three-dimensional models [Wang 2017] [Maturana 2015]. This kind of method can directly use 3D convolution to convolute voxels. However, voxels are not a good way to express three-dimensional models. The resolution of three-dimensional models expressed by voxels is not high. If we want to improve the resolution, we need to take up a lot of storage space.

Another method is based on picture groups [Su 2015]. Based on the method of picture group, the images from different angles are convoluted for 2D and merged into the classification layer. This method is not strictly based on three-dimensional models and requires a lot of pre-operation.

2.2 Point Cloud Method

Because point cloud can be used to express the three-dimensional model effectively, many researchers are studying the classification method based on point cloud now. The main difficulty of point cloud-based method is the disorder of point cloud. PointNet [Qi 2016] using symmetry function to solve disorder problems. PointCNN [Li 2018] explored the idea of equivariance instead of invariance. They train the possible sequence of point cloud directly.

We have come up with completely different ideas. Considering that point clouds belong to space coordinates, there always exists a function to abstractly express point clouds when they are placed on coordinate axes. As long as we can find this function, point clouds can be classified according to the output of the function. Of course, this function may be very complex, but it doesn't matter. Deep learning tools are very suitable for solving this problem.

Contact: Lu SUN, CHIBA University, 1-33, Yayoicho, Inage Ward, Chiba-shi, Chiba, 263-8522 Japan, 080-3974-0897, yeelou@chiba-u.jp

3. Local Feature Fitting Learning Network

Before introducing the network structure, I would like to review how convolution works in images. Example, computing any pixel on an image with convolution kernel 3×3. The first step is to find eight points adjacent to the pixel, then multiply the nine pixels with the corresponding position of the convolution core, and finally add up all the results. From the point cloud-based convolution operation, finding neighborhoods and building convolution kernels are the key steps for point clouds. So, we divide the network into four blocks according to the functional requirements. First resampling block is to find the neighborhood. Transform block and local feature fitting block are to determine the convolution kernels. The last block is to classify the extracted features. The network structure is shown in Figure 1.

3.1 Resampling

The purpose of this section is to resample the point cloud and make it look like many small point groups. It is required here that for a large point cloud, the number of small point groups is fixed, and the number of points of small groups is fixed.

Firstly, we sampled uniformly in point clouds. To get the core of the point group and then search the neighborhood based on these cores to get the point group.

As shown in Figure 1.A, the point cloud format changes from $n \times 3$ to $pg \times pn \times 3$. Where n is the point's number, pg is point group's number, pn is point number of point group.

3.2 Transform

The purpose of building this block is that the function expression does not have rotation and translation invariance for point clouds.

Although the center of the whole point cloud is at the origin of the coordinates, however, due to resampling, all point groups need to be standardized first.

$$x_{std}^{(i)} = \frac{x_i - \mu_x}{\delta_x}$$

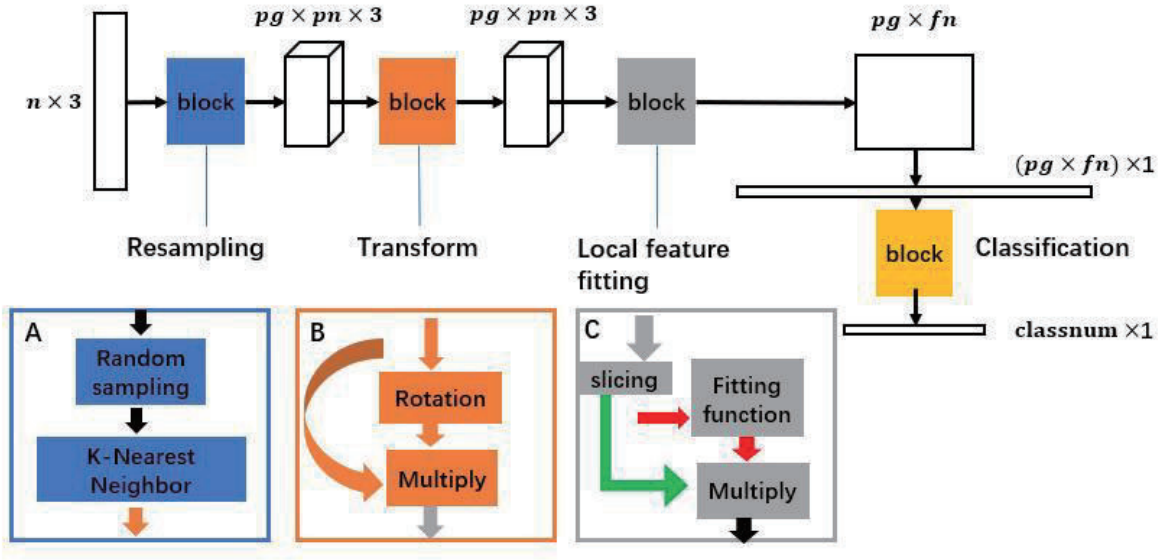

Fig 1. Local feature fitting learning network (LFFLN) architecture

Where $x_{std}^{(i)}$ is a normalized feature, x_i is an input feature, μ_x is the mean of the features, δ_x is the standard deviation of features.

Then we calculate the rotation matrix by regression of the point group itself. As shown in Figure 1.B, we only use simple non-linear regression, superimpose two dense layers, and then multiply this 3×3 matrix with itself to get the rotated point group. The input and output format of this block is unchanged.

3.3 Local Feature Fitting

The main purpose of this block is to make a convolution kernel. Because point clouds of the same shape have a different distribution of points. To get the same feature, a function needs to be calculated. From this function, we can get convolution kernels of any fixed shape.

First, cut the input of this block.

$$(\mathrm{pg} \times \mathrm{pn} \times 3) \rightarrow M\colon (\mathrm{pg} \times \mathrm{pn} \times 2), N(\mathrm{pg} \times \mathrm{pn} \times 1)$$

M retains the first two values of the point group (x, y), N retains the last values of the point group (z).

Then we put M into the fitting network to calculate a value N' (z'), This value is a small part of the convolution kernel. By adding all Z members N, we get the convolution result of a point group. By multiplying N' and N separately and adding the results, we can get one convolution result of a group of points.

$$feature_{conv} = \sum_{pn} \mathrm{F}(M) \times N$$

Where F( ) is a fitting function. We can construct multiple functions to obtain multiple features. Like the transform block, the function consists of two simple dense layers.

As shown in Figure 1.C, the input format changes from $\mathrm{pg} \times \mathrm{pn} \times 3$ to $\mathrm{pg} \times \mathrm{fn}$ where fn is the function's number.

3.4 Classification

There's nothing special about this piece. Pull the input into a one-dimensional vector and put into three layers of dense to get the final classification result.

3.5 Loss

The loss function of this method is based on cross-entropy loss. However, there is a rotating block before, and the network cannot converge very well. To guide the network learning, we will transform the point group by SVD before input the rotating block. We will calculate mean squared error loss by rotating the characteristic matrix and the result of the rotating block.

$$\mathrm{Loss} = loss_{ce} + \alpha loss_{rmse}$$

Where α is the weight of $loss_{rmse}$. As the training progresses, it gradually decreases.

4. Experiment

Our network learns a global point cloud feature that can be used for object classification. We evaluate our model on the ModelNet40 [Wu 2015] shape classification benchmark. The data set split into 9,843 for training and 2,468 for testing.

In Table 1, we compare our model with previous works; Our model achieved excellent performance among methods based on 3D input (volumetric, image and point cloud).

Table 1: Object classification results on ModelNet40

Method	Input	accuracy avg. class	accuracy overall
3DShapeNets	volume	77.3	84.7
VoxNet	volume	83.0	85.9
MVCNN	image	90.1	-
PointNet	point	86.2	89.2
PointCNN	point	88.1	91.2
Ours	point	86.9	88.1

We input the result of feature fitting directly into the classification part. This results in that the classification results are based on the local features of point clouds. However, the types of point clouds in this data set vary greatly. So although it is based

on the classification of local point clouds, good results are obtained.

5. Discuss

We want to discuss some hyperparameters. In Section 3.1, pg and pn are the number of point groups and the number of points in point group. In an image, the stride of the general convolution operation is 1. For 3×3 convolution kernel, each operation overlaps by 66%. So we sampled four adjacent points evenly in the point cloud and then set 27 adjacent points as a group of points, $\mathrm{pg} = 512$, and $\mathrm{pn} = 27$.

We calculate the rotation matrix by regression of the point group itself in Section 3.2. Considering that there are only 27 points in a point group, so we don't need too complicated network model to return to a 3×3 matrix. We superimpose two layers of dense at first. But the rotation matrix is a very complicated process. Although we add an item to the rotation matrix independently in the loss function, when we check the training results, we find that this part of the network can not converge very well. Therefore, in the follow-up study, we consider increasing the complexity of this part of the network and improving the overall accuracy.

After resampling, feature rotation and feature fitting, we can think that this is a convolution operation for point clouds. In this study, these features are directly put into the classification network for classification. We know that one-layer convolution is not enough for image classification, so if the feature is sampled, rotated and fitted continuously, the classification accuracy can be improved. Attention should be paid to the convergence of the rotation matrix for features. It is not possible to add a function on this part of the loss function. How to train this part of the network is the focus of future research.

Although we are not the most precise model, we still have a lot to improve. For example, we have performed a convolution operation now. If we add more convolution operations later, I believe the result will be better.

6. Conclusion

We have designed a point cloud feature extraction method by simulated convolution, which achieves good results when only one layer of simulated convolution is used, and no tricks are used. The plan is to add more simulated convolution layer and use various tricks to improve accuracy.

References

[Surname of the first author Year] Names of authors, Title, Journal, Publisher, Year.

[Wang 2017] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-CNN: Octree-based convolutional neural networks for 3d shape analysis. ACM Transactions on Graphics, 2017

[Maturana 2015] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In IEEE/RSJ International Conference on Intelligent Robots and Systems, 2015

[Su 2015] H. Su, S. Maji, E. Kalogerakis, and E. G. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In IEEE International Conference on Computer Vision, 2015

[Qi 2016] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. Computing Research Repository - arXiv, 2016

[Li 2018] Y. Li, R. Bu, M. Sun, and B. Chen. Pointcnn. arXiv:1801.07791, 2018

[Wu 2015] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015