ENRICH FEATURES FOR FEW-SHOT POINT CLOUD CLASSIFICATION

Hengxin Feng   *Weifeng Liu*   *Yanjiang Wang*   *Baodi Liu*[*]

College of Control Science and Engineering, China University of Petroleum (East China)

ABSTRACT

Recently, many existing fully supervised methods for point cloud classification have strongly promoted the development of point cloud learning. However, these methods require a lot of labeled data as support, which is challenging to obtain. To alleviate this problem, we propose a novel few-shot point cloud classification method to classify new categories given a few labeled samples. Specifically, we apply the feature supplement module to enrich the geometric information of points and then aggregate multi-scale features through the channel-wise attention module while reducing the computational complexity. Finally, we introduce a classifier to classify the point cloud features under the few-shot learning setup to predict its label. We carry out experimental verification on the benchmark dataset and achieve state-of-the-art performance.

Index Terms— Point cloud, Few-shot classification, Channel-wise attention

1. INTRODUCTION

With the development of automatic driving and robot interaction technology, 3D data processing has attracted more and more attention in computer vision. Point cloud data is one of the most widely used 3D data representations, which can be obtained in real-time by scanners such as lidar, so it is particularly suitable for end-to-end vision tasks. Point cloud classification is a fundamental 3D computer vision problem, which aims to predict the category of each point cloud. However, it is challenging to deal with point clouds because of their irregularity and disorder.

Before the creation of point-based methods, many researchers processed point clouds based on multi-view or voxelization. MVCNN [1] projected the point cloud into multiple 2D images and employed the maximum pooling operation to aggregate the features of each view. Yang *et al*. [2] obtained a three-dimensional object representation by gathering the relationships between regions and views on a set of views. View-GCN [3] introduced a directed graph and applied graph convolution to improve recognition accuracy. VoxNet [4] voxelized the point cloud into a three-dimensional grid and used three-dimensional convolution to classify it. To reduce the computational complexity of voxelization, OctNet [5] introduced an octree structure to process point clouds.

The method of projecting the point cloud into multiple images or voxelization is expensive, which promotes the birth of point-based methods. PointNet [6] innovatively adopted the original point cloud data as input and used symmetric functions to deal with the disorder of the point cloud, which became the originator of the point-based method. In order to solve the problem that PointNet cannot capture the local structure information of the point cloud, the hierarchical network PointNet++ [7] is proposed. [8, 9, 10] defined the convolution kernel on a continuous space or a regular grid. The spatial distribution or offset of adjacent points relative to the center point determines its weight to apply convolution to the point cloud. Aiming at the spatial characteristics of the point cloud, DGCNN [11] introduced the graph, regarded each point in the point cloud as a vertex of the graph, and then extracted the features among points through the edge convolution operation. LDGCNN [12] generated the directed edge of the graph according to the neighboring points and performed feature learning in the spatial domain. In the spectral domain, Point-GCN [13] defined convolutional filters as Chebyshev polynomials and used Gaussian kernel to weight each edge.

Although these fully supervised methods have achieved excellent performance on the benchmark of public datasets such as ModelNet40 [14] and ScanObjectNN [15], they seriously rely on enormous labeled point cloud data. Collecting these data is time-consuming and laborious. To combat this problem, SS-FSL [16] proposed a hierarchical self-supervised point cloud learning approach based on cover-tree. However, it is challenging to effectively combine with the current state-of-the-art supervised methods. This paper proposes a novel few-shot point cloud classification method to alleviate network dependence on labeled data, which can effectively combine various supervised approaches. Specifically, we first use the point-based embedding network to extract the point cloud features. We introduce the feedback structure to modify the learning process and aggregate multi-scale features through channel-wise attention. After extracting features through the embedding network, a point cloud is represented as a feature

This research was supported by the Natural Science Foundation of Shandong Province under Grants ZR2019MF073, the Fundamental Research Funds for the Central Universities, China University of Petroleum (East China) under Grant 20CX05001A, the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-008, and the Creative Research Team of Young Scholars at Universities in Shandong Province under Grant 2019KJN019. *Corresponding author.

978-1-6654-0540-9/22/$31.00 ©2022 IEEE

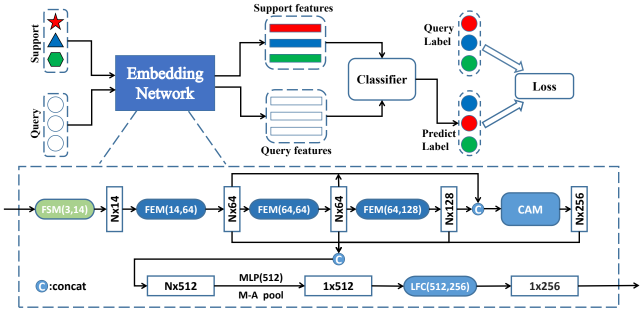

Fig. 1. The architecture of our method is mainly composed of embedding network, classifier, and loss function. The feature supplement module provides rich geometric clues for feature learning of cascaded FEMs, and CAM aggregates multi-level features. Then we combine max-pooling, average-pooling, and fully connected layers to refine the concatenated features, which are further sent to the classifier for classification.

vector. In this process, we apply a training class set to train the parameters of the embedding network. Then we divide the testing class set into support set and query set and learn their high-dimensional feature representation through the network. Finally, we introduce a classifier to predict the categories of the query set features according to the features and labels of the support set. Our main contributions are summarized as follows:

- We propose a novel few-shot point cloud classification paradigm, which can effectively combine the current fully supervised classification methods.
- We enrich point cloud features through feature supplement module and introduce feedback mechanisms and channel-wise attention to improve the accuracy of feature learning.
- We obtain the state-of-the-art experimental results on the benchmark dataset, which validates the effectiveness of our proposed method.

2. PROPOSED METHOD

In this section, we introduce the proposed algorithm for few-shot learning on point clouds, which uses the feedback feature extraction module and channel-wise attention to improve the performance of the few-shot classification task.

2.1. Problem Definition

We follow the episodic paradigm of few-shot classification tasks on images and combine it with the training and testing of point cloud classification. Specifically, we divide the dataset into training class set D_{train} and testing class set D_{test} according to categories, where $D_{train} \cap D_{test} = \emptyset$. We input point cloud data into the model in the form of tasks. Each few-shot task can be instantiated as an N-way K-shot episode, containing a pair of support and query sets. K samples are drawn from each of the N randomly selected classes to form the support set, denoted as $S = \{(P_s^1, L_s^1), (P_s^2, L_s^2), \cdots, (P_s^{N\times K}, L_s^{N\times K})\}$. Then T samples of each class are extracted from the remaining samples to form a query set, denoted as $Q = \{(P_q^1, L_q^1), (P_q^2, L_q^2), \cdots, (P_q^T, L_q^T)\}$. The goal of few-shot point cloud classification is to learn a model $M_\Phi(Q, S)$ from S and correctly predict the labels of query samples in Q.

2.2. Embedding Network

The embedding Network is the most critical part of the entire algorithm, and the features it learns directly affect the classification performance of subsequent classifiers. The bottom part of Fig 1 shows the three main components of the embedding network: feature supplement module (FSM), feedback feature extraction module (FEM), and channel-wise attention module (CAM).

Feature supplement module. Generally, the point cloud data we obtain contains only a few basic features $\mathbb{R}^3 : (x, y, z)$ or $\mathbb{R}^6 : (x, y, z, r, g, b)$. The feature supplement module aims to provide the geometric relationship information among the points of the original point cloud for feature learning in high-dimensional space. Specifically, for each point $p_i \in \mathbb{R}^3$ in the original point cloud, we first search its two nearest neighbors $p_j^1, p_j^2 \in \mathbb{R}^3$. Then we use these two points to supplement the original point $p_i \in \mathbb{R}^3$ into a new point $p_i^{'} \in \mathbb{R}^{14}$.

$$\begin{aligned} p_i^{'} =& [p_i,\ p_j^1 - p_i,\ p_j^2 - p_i,\ \left|p_j^1 - p_i\right|, \\ & \left|p_j^2 - p_i\right|,\ (p_j^1 - p_i) \times (p_j^2 - p_i)] \end{aligned} \tag{1}$$

The feature supplement module replenishes three features to each point: a normal vector and the relative position and distance between the center point and two neighbors. The relative position and distance between points can implicitly provide the distribution density and position information of local points. The normal calculated by the cross product of two relative position vectors can improve the robustness of features in the case of point cloud scaling, translation and rotation. The feature supplement module enriches the features of the points in the original point cloud, which provides more geometric clues for feature learning in advanced spaces.

Feedback feature extraction module. Inspired by GBNet [17], we introduce the feedback feature extraction module, as shown in Fig 2. The module adjusts the input features through the feedback learned features to promote the network to learn more appropriate parameters. Specifically, for the input feature map $f_{in} \in \mathbb{R}^{N\times d}$, query the k nearest neighbors x_j^k of each point x_i through the k-nearest neighbors(k-NN) algorithm, and then translate them to the spatial coordinate system with the query point as the origin to obtain the position $(x_j^k - x_i)$. Concatenate the features of query point and transformed points to obtain the aggregate features $(x_i, x_j^k - x_i)$, through the MLP aggregation feature obtain f_{AG}. We denote the above operation as EdgeConv.

$$f_{AG} = \text{EdgeConv}_1(f_{in});\ f_{AG} \in \mathbb{R}^{N \times d^{'} \times k} \quad (2)$$

In order to apply the feedback mechanism, we use an MLP layer to back-project the aggregated features to the initial dimension and make the difference from the original input to obtain the feedback signal. Then another EdgeConv is introduced to extract the features from the feedback signal. We define the feature map after secondary learning as f_{FB}:

$$f_{FB} = \text{EdgeConv}_2(MLP(f_{AG}) - f_{in});\ f_{FB} \in \mathbb{R}^{N \times d^{'} \times k} \quad (3)$$

After obtaining the two aggregated feature maps f_{AG} and f_{FB}, the max pooling operation is applied to extract the salient features. Then place a channel attention module to refine the final feature representation. Generally, the output of the feedback feature extraction module is f_{out}:

$$f_{out} = CAM\left(Max_{\{k\}}(f_{AG} + f_{FB})\right);\ f_{out} \in \mathbb{R}^{N \times d^{'}} \quad (4)$$

Channel-wise attention module. Although the self-attention mechanism can refine the feature representation of each point by aggregating the features of all other points in the point cloud, it also greatly increases the computational complexity of the network. As shown in Fig 3, we introduce the channel attention module. The module distributes attention weights along the channel and reduces the number of points from N to $N^{'}(N^{'} < N)$ to reduce channel redundancy and computational cost.

Specifically, we apply three independent MLP operations to generate Query Matrix: $\mathcal{Q}$, Key Matrix: $\mathcal{K}$ and Value Matrix: $\mathcal{V}$. Then the transposed Query Matrix and Key Matrix are generated attention map by the product operation: $\mathcal{A}_{d \times d} = \mathcal{Q}^T \mathcal{K}$, where $\mathcal{A}_{m,n}$ on behalf of the attention relationship between the m^{th} channel and the n^{th} channel of the input feature map. After the operation of Softmax, the attention matrix is multiplied by Value Matrix to calculate each channel of the point feature. In addition, we use a residual connection to avoid vanishing gradients and improve training efficiency. α is a learnable weight parameter. Overall, for the input feature map $f_{in} \in \mathbb{R}^{N \times d}$, the output of CAM is:

$$f_{out} = CAM(f_{in}) = \alpha \mathcal{V} \mathcal{A} + f_{in};\ f_{out} \in \mathbb{R}^{N \times d} \quad (5)$$

2.3. Classifier classification

After passing through the network, the point clouds of the support set and query set are represented as feature vectors.

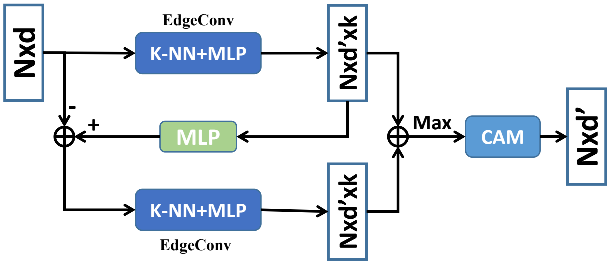

Fig. 2. **FEM**: Feedback feature extraction module. This module implements feature feedback through EdgeConv and MLP, and finally refines features through CAM.

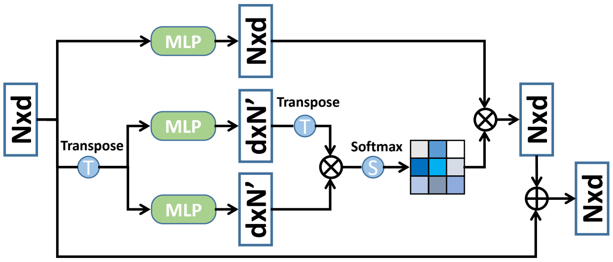

Fig. 3. **CAM**: Channel-wise attention module. Based on self-attention, the channel-wise attention module reduces the computational complexity while capturing global information.

We input these features into the classifier to predict the label of the query set point cloud. After normalizing the prediction result into a probability distribution [18], we can compute the cross-entropy loss between the distribution and the ground truth labels.

3. EXPERIMENTAL RESULTS

In this section, we first explain the training and testing procedure and parameter settings of the proposed network in detail. Then we compare our method with the state-of-the-art research on a general point cloud classification dataset. Furthermore, we conduct related ablation experiments to analyze the influence of different components of the model and demonstrate the effectiveness of our method.

3.1. Experimental Setup

Implementation details: For simplicity, we only use the coordinates (x, y, z) of the point as the original input to the network. We first use the point cloud data in training set D_{train} to train the embedding network. After FSM supplements the point cloud features, three feedback feature extraction modules with output feature scales of $(64, 64, 128)$ are used to learn advanced features. The number of k-NN nearest neighbors is $K = 20$. We set the batch size to 32, and the number of the epochs is 200. We use the SGD optimizer with a momentum of 0.9, and weight decay is 0.0001. The learning rate is initially set to 0.1 and reduced to as low as 0.001 through the cosine annealing algorithm. Then we test the classification performance of the model for the new class of data in the few-shot task on D_{test}. Our model is trained on Linux and Nvidia Tesla V100 GPU with PyTorch 1.7.1.

Dataset: ModelNet40. Due to its authority, this dataset is the most widely used in point cloud classification. It consists of $12,311$ CAD-generated meshes from 40 categories. In order to follow the few-shot learning setup, we divide the dataset by category. 24 categories are for network training, and the remaining 16 categories are for few-shot testing. In the experiment, we uniformly sample $1,024$ points on the surface of each CAD model, and each point only considers three-dimensional coordinates as its initial features.

3.2. Experimental Results

Table 1 shows the few-shot point cloud classification results on ModelNet40. Compared with the most advanced methods, our proposed method has significantly improved the accuracy. The point cloud used in the experiment only contains $1,024$ points, and each point only contains three-dimensional coordinate information as the original feature. The result is an average of 300 episodes results. Note that the results of all other networks in the table are from SS-FSL [16]. In terms of the result, our approach is indicated to be effective and promising.

Table 1. Classification results on ModelNet40 using accuracy metric (%) for few-shot learning setup. The results of other methods are from SS-FSL.

Method	5-way		10-way	
	10-shot	**20-shot**	**10-shot**	**20-shot**
PointCNN[8]	65.41	68.64	46.60	49.95
PointNet[6]	51.97	57.81	46.60	35.20
PointNet++[7]	38.53	42.39	23.05	18.80
DGCNN[11]	31.60	40.80	19.85	16.85
Latent-GAN[19]	41.60	46.20	32.90	25.45
3D-GAN[20]	55.80	65.80	40.25	48.35
SS-FSL[16]	63.20	68.90	49.15	50.10
Ours	**76.69**	**85.76**	**68.76**	**80.72**

Table 2. Results on ModelNet40 for few-shot learning setup of different classifiers.

Method	3-way 1-shot	5-way 1-shot
RandomForest Classifier	58.07	49.11
SVM(kernel=linear)	76.82	66.80
Logistic Regression	76.83	67.26
SVM(kernel=rbf)	**77.16**	**67.38**

Table 3. Ablation study about different modules on ModelNet40. FSM:Feature supplement module, FEM:Feedback feature extraction module, CAM:Channel attention module.

FSM	FEM	CAM	5-way 1-shot
-	√	√	65.89
√	-	√	63.37
√	√	-	66.24
√	√	√	**67.38**

3.3. Ablation Studies

In order to verify the influence of different classifiers on the model results, we conducted comparative experiments on ModeNet40 through different few-shot settings. As shown in Table 2, the support vector machines(SVM) based on the Gaussian kernel function performs best. We analyze that because the few-shot classification task has the characteristics of fewer samples and more features, the Gaussian kernel can map each example to a new feature space that is more conducive to measuring the similarity between samples. In this space that characterizes similarity, instances of the same class can be better clustered together and linearly separable.

To verify the effectiveness of the different modules in the embedding network, we performed ablation studies on the components introduced by our network. Table 3 shows the results of the ablation study concerning the effects of different components in the embedding network. Among them, when not using FEM, we apply an EdgeConv instead. We also implemented an ablation experiment on the CAM module before the pooling operation to verify its role in aggregating features at various levels. The table shows that the extended features of our feature supplement module can facilitate subsequent feature learning in high-dimensional spaces for better results. The introduction of FEM and CAM modules can increase the classification accuracy by 4.01% and 1.14%. The final result shows the effectiveness of each module and the flexibility of our proposed method.

4. CONCLUSION

This paper combines the point-based method with few-shot learning, designs a novel few-shot point cloud classification paradigm. It effectively combines the existing fully supervised methods so that the model can distinguish the point cloud in the new class through a few labeled samples. Besides, we supplement the original point cloud features and apply feature feedback and channel attention to comprehensively learn the local and global information of the point cloud. The experimental results on the benchmark dataset prove the effectiveness of our method. In the future, we plan to optimize the model and extend the approach to tasks such as point cloud semantic segmentation under few-shot settings.

5. REFERENCES

[1] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller, “Multi-view convolutional neural networks for 3d shape recognition,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 945–953.

[2] Ze Yang and Liwei Wang, “Learning relationships for multi-view 3d object recognition,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7505–7514.

[3] Xin Wei, Ruixuan Yu, and Jian Sun, “View-gcn: View-based graph convolutional network for 3d shape analysis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1850–1859.

[4] Daniel Maturana and Sebastian Scherer, “Voxnet: A 3d convolutional neural network for real-time object recognition,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 922–928.

[5] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger, “Octnet: Learning deep 3d representations at high resolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3577–3586.

[6] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[7] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[8] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen, “Pointcnn: Convolution on x-transformed points,” in *Advances in Neural Information Processing Systems*, 2018, vol. 31, pp. 820–830.

[9] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas, “Kpconv: Flexible and deformable convolution for point clouds,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6411–6420.

[10] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan, “Relation-shape convolutional neural network for point cloud analysis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8895–8904.

[11] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon, “Dynamic graph cnn for learning on point clouds,” in *Acm Transactions On Graphics*, 2019, vol. 38, pp. 1–12.

[12] Kuangen Zhang, Ming Hao, Jing Wang, Clarence W de Silva, and Chenglong Fu, “Linked dynamic graph cnn: Learning on point cloud via linking hierarchical features,” *arXiv preprint arXiv:1904.10014*, 2019.

[13] Yingxue Zhang and Michael Rabbat, “A graph-cnn for 3d point cloud classification,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6279–6283.

[14] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao, “3d shapenets: A deep representation for volumetric shapes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[15] M.A. Uy, Q.H. Pham, B.S. Hua, T. Nguyen, and S.K. Yeung, “Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data,” in *2019 IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1588–1597.

[16] Charu Sharma and Manohar Kaul, “Self-supervised few-shot learning on point clouds,” in *Advances in Neural Information Processing Systems*, 2020, vol. 33, pp. 7212–7221.

[17] Shi Qiu, Saeed Anwar, and Nick Barnes, “Geometric back-projection network for point cloud classification,” *IEEE Transactions on Multimedia*, 2021.

[18] Na Zhao, Tat-Seng Chua, and Gim Hee Lee, “Few-shot 3d point cloud semantic segmentation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8873–8882.

[19] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T Freeman, and Joshua B Tenenbaum, “Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling,” in *Advances in Neural Information Processing Systems*, 2016, pp. 82–90.

[20] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas, “Learning representations and generative models for 3d point clouds,” in *International conference on machine learning*, 2018, pp. 40–49.