Applied Intelligence (2022) 52:14787–14798
https://doi.org/10.1007/s10489-021-02840-2

Multi-view attention-convolution pooling network for 3D point cloud classification

Wenju Wang[1] · Tao Wang[1] · Yu Cai[1]

Accepted: 11 September 2021 / Published online: 30 October 2021

Abstract

Classifying 3D point clouds is an important and challenging task in computer vision. Currently, classification methods using multiple views lose characteristic or detail information during the representation or processing of views. For this reason, we propose a multi-view attention-convolution pooling network framework for 3D point cloud classification tasks. This framework uses Res2Net to extract the features from multiple 2D views. Our attention-convolution pooling method finds more useful information in the input data related to the current output, effectively solving the problem of feature information loss caused by feature representation and the detail information loss during dimensionality reduction. Finally, we obtain the probability distribution of the model to be classified using a full connection layer and the softmax function. The experimental results show that our framework achieves higher classification accuracy and better performance than other contemporary methods using the ModelNet40 dataset.

Keywords 3D point cloud · Multi-view · Attention-convolution pooling · Point cloud classification

1 Introduction

The growth in the number of 3D cameras, Kinect-type devices, radar, depth scanners, and other 3D camera and scanning devices has improved the collection and accuracy of point cloud data. Point cloud data have been widely used in for self-driving vehicles [1], intelligent robots [2, 3], virtual reality [4], medical diagnoses [5], and medical imaging [6]. Among all types of cloud point processing, classification is the basis for target recognition and tracking [7], scene interpretation [8], and 3D reconstruction [9]. There is thus tremendous research significance to 3D point cloud classification.

Point cloud classification using traditional machine learning methods [10] suffers from long training times and poor classification accuracy. The rapid development of deep learning technology [11] over the past decade coupled with the emergence of 3D model datasets (such as ShapeNet [12], ModelNet [13], PASCAL3D+[14], and the Stanford Computer Vision and Geometry Laboratory datasets) has resulted in new approaches to classifying 3D point cloud data. The purpose of our research is to improve the accuracy of point cloud classification.

Existing deep learning methods for classifications fall into three categories depending on the sort of convolution object used: voxel-based methods, point cloud-based methods, and view-based methods [15]. Among them, the voxel-based method transforms the point cloud into volume pixel with fixed size and employs convolution neural network for classification. For the point cloud-based method, the point cloud is directly input into the neural network to complete the classification. The view-based method converts 3D point cloud into 2D images from different perspectives, which makes the classification problem of 3D point cloud become 2D image classification.

1.1 Voxel-based method

Maturana et al. [16] introduced the VoxNet model that converts 3D data into regular 3D voxel data and then uses a 3D convolutional neural network (CNN) to extract spatial local correlations of 3D voxel data to perform classification. Wang et al. [17] proposed an octree data structure that uses simple calculations

✉ Tao Wang
193712574@st.usst.edu.cn

Wenju Wang
wangwenju@usst.edu.cn

Yu Cai
192402566@usst.edu.cn

[1] College of Communication and Art Design, University of Shanghai for Science and Technology, Shanghai, China

to quickly compute the index values for storing planar information and limiting the planar computations to the vicinity of the planar, reducing computing overhead. Brock et al. [18] presented a voxel-based deep CNN for classification, improving the classification data of the model network significantly and solving the unique challenges of voxel-based representation.

However, effective voxel-based methods are usually limited to small datasets or single-object classification and are computationally expensive. On large datasets, the accuracy of these methods is poor. Therefore, there is still much room for improvement for these methods.

1.2 Point cloud-based method

Point cloud-based methods have been the primary tool for 3D object classification. The most representative of these are PointNet [19] and PointNet ++ [20] proposed by Qi. The PointNet method uses two Spatial Transformer Networks (STNs) to solve the rotation problem. PointNet++ learns the point cloud characteristics using hierarchical features. Increasing the depth of the network layers produces more accurate local features, but its complex architecture has a high computational cost. Klokov et al. [21] proposed a K-d tree-based deep network called KD-Net and tested it with a ModelNet [9] dataset. KD-Net first uses the K-d tree structure to create point clouds in a certain order and then shares the weight attributes of different tree structures. After calculating the root node characteristics from the bottom-up, the whole point cloud is sent to the fully-connected layer for classification prediction. This method is a classical deep learning point cloud-oriented approach and partially classifies point clouds. However, it is sensitive to noise and requires training a new model for each point cloud input, making both calculation and training difficult. Riegler et al. [22] proposed Oct-Net, a deep learning method using sparse 3D data, enabling CNNs to have more levels and higher resolution. Li et al. [23] proposed a simple and universal PointCNN point cloud feature learning framework with good performance on many challenging tasks. Wang et al. [24] proposed an EdgeConv layer to acquire local features, solving the local feature problem unaddressed by PointNet. This method uses 3D point clouds as input data and uses dynamic convolution to extract the 3D point cloud features. The PointGrid model proposed by Le et al. [25] better represents the details of local geometric shapes. PointGrid uses an embedded voxel grid with a regular structure, enabling 3D volume integration to extract global information hierarchically and to perform well even without a high-resolution grid. Therefore, PointGrid is simpler and faster to train and test. PointGrid also uses less memory than Oct-Net, PointNet, PointNet++, and Kd-Net.

However, point cloud-based methods are unable to adapt to non-uniform point sampling density [26], leaving the classification accuracy in need of improvement. In addition, the inability of such methods to determine the exact location of discrete objects is a major limitation.

1.3 View-based method

To classify 3D point clouds, the models represent views from multiple perspectives and process them in parallel using CNNs, making such multi-views methods efficient. Gao et al. [27] proposed an algorithm framework CCFV for unconstrained views, in which each object is represented by a set of free views that are able to capture objects in any direction without camera constraints. Although this framework has good performance, it loses significant information when capturing any direction. Su et al. [28] proposed a method to extract the specific combination of 3D shape descriptors by rendering a single 3D shape into images from different perspectives. This method is called MVCNN. However, this method does not effectively combine the feature relationships between multi-view, limiting the distinguishability of the final feature descriptor.

Many effective view-based methods [29–34] have emerged based on MVCNN. They use views from multiple perspectives to represent 3D models. We will elaborate on related work of view-based method in Section 2. They typically achieve higher classification accuracy with fewer computational requirements, offering better classification performance than voxel-based and point cloud-based methods, but they lose some details in the representation or processing of views. Thus, the accuracy of these algorithms is not very high, with significant room for improvement.

1.3.1 Motivation

Our chief motivation is to improve the accuracy of point cloud classification by multi-view 3D point cloud classification algorithms. We need to consider the loss of information caused by feature representation and the loss of detail for each view in the process of dimension reduction.

With regard to information loss with feature representation, traditional CNN models extract single-view visual features and focus on information fusion, but they ignore the loss of feature information. GaitSet [35] used an attention mechanism in the pooled layer to illustrate that the attention mechanism effectively solves this problem in the pooled layer. However, this method extracts spatial and temporal information, and is not directly suitable for our problem.

With regard to information loss during dimensional reduction, traditional convolution pooling methods expose more information but lose some details during dimension reduction, leaving only the information considered important. Traditional convolution pooling methods are unsuited to our problem as some details also affect the classification accuracy of 3D point clouds.

To solve these problems, we propose a multi-view attention-convolution pooling network (MVACPN) method for 3D point cloud classification. Compared with traditional methods, our approach uses the attention-convolution operation to find useful information related to the current output in the input data, effectively resolving the loss of feature information caused by the feature representation and views in the process of dimension reduction, so as to improve the classification accuracy.

1.3.2 Contributions

We summarize our contributions as follows.

(1) To improve the accuracy of view-based 3D classification tasks, we propose a multi-view attention-convolution pooling network (MVACPN) for 3D point cloud classification. By introducing the Res2Net [36], a variant of ResNet [37], we extract features from a set of 2D images, further improving the accuracy of 3D model classification tasks.

(2) To improve the classification accuracy of the neural network, we propose an attention-convolution feature pooling method. Using the attention-convolution mechanism, we focus more on finding useful information related to the current output in the input data for processing, effectively solving the loss of feature information caused by feature representation and the loss of detail information in each view during dimension reduction. This improves the accuracy of classification.

(3) We performed a number of experiments to verify the performance of the proposed method. Compared with several popular methods, the experimental results show that the accuracy of our classification method is higher, reaching 93.64%, showing that our classification framework achieves advanced performance.

We organize our paper as follows. In Section 2, we introduce in detail the related work of view-based method. In Section 3, we describe our method. In Section 4, we present our experiment and its results. Finally, in Section 5, we present our conclusions and suggestions for future work.

2 Related work

Multiple views present the same object from multiple angles, which contains richer feature information than the traditional single view. How to use the complementary and compatible information of these views to give full play to their respective advantages and avoid their respective limitations, so as to obtain the most profound understanding of the target object. Therefore, we introduce some related work detailing multi-view processing tasks.

Wang et al. [38] developed a graph-based system (GBS), which is a general multi-view clustering system. Based on this, the effects of different graph metrics on multi-view clustering were discussed. Then, a new GBS-based multi-view clustering method was developed to overcome the shortcomings of clustering methods. This method can generate a uniform graph matrix after each SIG matrix is automatically weighted, and the final cluster is directly generated on the graph matrix. The final experiment showed that GBS has good robustness and effectiveness. Xiao et al. [39] constructed a recommendation model based on multi-view regular learning, which is an integration of multi-view data sources and can use the inherent structure of space for model learning. Zhang et al. [40] introduced a multi-task multi-view clustering algorithm based on locally linear embedding (LLE) and Laplacian eigenmaps (LE). This algorithm combines the advantages of LLE and LE. First, the multi-view samples are mapped to the view space, thus maintaining the relationship between the views in the same task. Then, the view space of the samples is converted to the sample space to the extent that one can learn the sharing and complementarity characteristics of multitask multi-view; however, this method requires additional clustering steps. Zhang, Yang et al. [41] designed a multi-view clustering algorithm based on non-negative matrix factorization, which can make full use of limited images to obtain the characteristics of the data and can handle the similarity relationship well between different objects. It successfully solved the disadvantage of setting parameters when exploring multi-view.

However, the above method [38–41] belongs to the multi-view clustering problem. For feature extraction and fusion of views, Hayashi et al. [42] presented a one-class classification model with high performance and low complexity, which uses the convolutional image transformation network to convert input images into target images and avoids the output diversity of the classification network and the process of extracting features extensively. Wu et al. [43] developed a multi-layer fusion framework based on point cloud features that can fuse local features of multi-convolution layer features to achieve global features. This method significantly improves the detection performance of small objects, but it requires accurate classification of point clouds before it can be used.

The most representative method of 3D point cloud classification based on multi-view is MVCNN, which was proposed by Su [28]. On the basis of MVCNN, Feng et al. [29] proposed their GVCNN framework, which extracts the visual characteristics of 2D images taken by 3D models from different perspectives, groups different feature subgroups, and then aggregates the visual features of each group for classification. Jiang et al. [30] proposed a multi-loop-view

CNN framework called MLVCNN, which generates cycle-level features for each view and considers the internal relationships of different views in the same cycle.

Nie et al. [31] proposed a multi-modal joint networks called MMJN to improve classification performance using the correlation between different features extracted from different modal networks. Yu et al. [32] proposed a multi-view harmonized bilinear network framework called MHBN taking full advantage of the relationship between its polynomial kernel and bilinear pool. Using the local convolution feature of the bilinear pool aggregation, it obtains an effective 3D object representation method. This method is more distinguishable.

Sun et al. [33] proposed the SRINET network framework. This approach uses point projection to derive rotationally invariant features, uses a PointNet-based backbone network to extract global information, and applies graphic aggregation to mine local shape features for point cloud data classification. However, the framework needs improvement in order to select a more stable axis to reduce the loss when transforming 3D coordinates into point projection features to further improve the classification accuracy. Zhou et al. [34] projected a 3D point cloud onto a 2D plane via a 360° projection. Using a CNN trained with only a unipolar view of the 3D shape, it obtains the polar view representation (PVR) to classify the 3D shape. However, the classification accuracy of this method for the ModelNet40 dataset is only 91.69%, and never exceeds 92%.

In order to further improve the classification accuracy of point clouds and to resolve the loss of feature information and detail information, we propose a multi-view attention-convolution pooling network for 3D point cloud classification. We will elaborate on our approach in Section 3.

3 Method

In this section, we detail our proposed method. Figure 1 shows our overall framework of MVACPN. We divide our algorithm into the multi-view generation module, the depth extraction visual feature module, and the visual feature fusion classification module. The multi-view generation module renders the original 3D point cloud model into a view with multiple perspectives. The deep extraction visual features module uses Res2Net to extract the visual features of each view. At the same time, the attention-convolution pooling process uses the attention mechanism to extract feature information from the view while the convolution operation extracts detailed information from the view. The visual feature fusion classification module uses the output of attention-convolution pooling to fuse and classify feature and detail information of visual features.

3.1 Multi-view generation module

For an original 3D point cloud model, we first render it with voxels [18] and then use a set of virtual images $V = \{V_1, V_2, V_3, \cdots, V_n\}$ from different perspectives to replace the virtual 3D model, where V_n represents n virtual images generated by a 3D model from n perspectives. For this process, we place n virtual cameras on a circular track at the same angle d of the interval and aim each virtual camera's capture lens at the center of the 3D model to simulate human observation of the model. The relationship between the distance angle d of the virtual camera, with n is $d = 360°/n$. For this article, we set up three types of virtual camera arrangements. The first uses three virtual cameras with an interval angle of $d = 120$ degrees to obtain three views. The second uses six virtual cameras, with interval angle $d = 60$ degrees to obtain six views as shown in Fig. 2a.

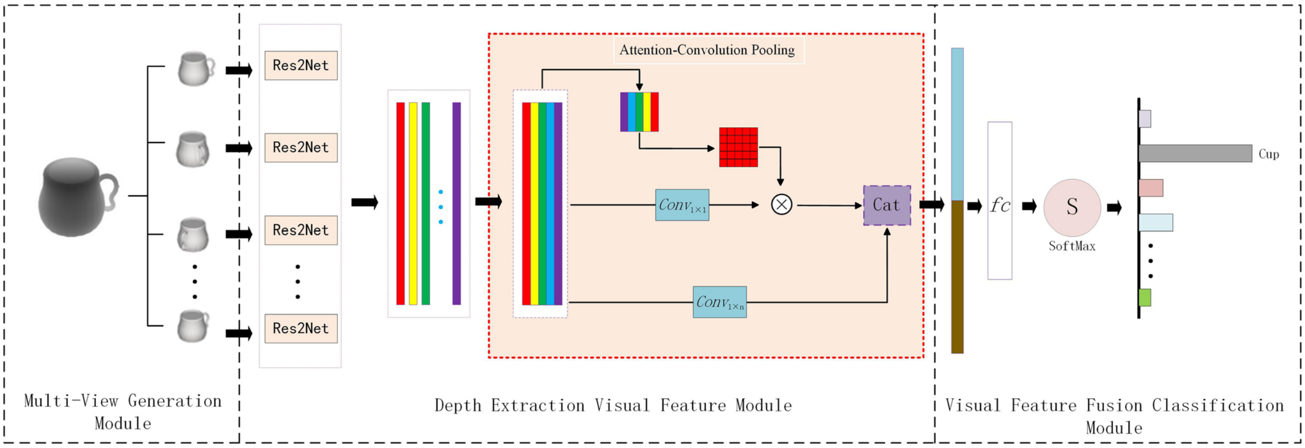

Res2Net
Res2Net
Res2Net
Res2Net
Attention-Convolution Pooling
$Conv_{1\times1}$
$Conv_{1\times n}$
Cat
fc
S
SoftMax
Cup
Multi-View Generation Module
Depth Extraction Visual Feature Module
Visual Feature Fusion Classification Module

Fig. 1 The framework of our approach

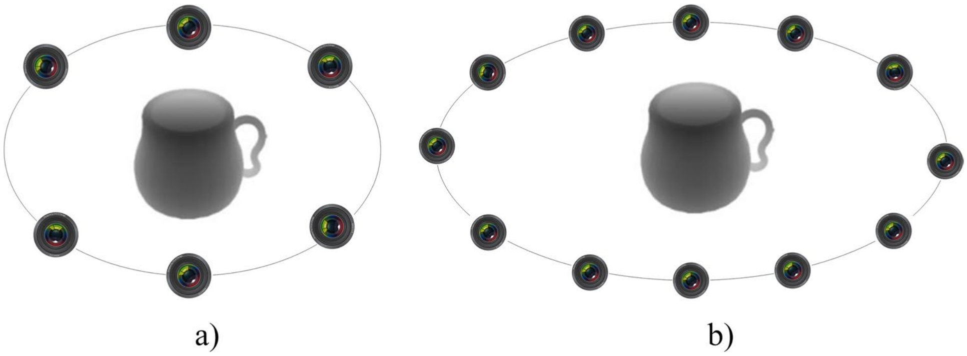


Fig. 2 Camera arrangements for 6-view and 12-view model images

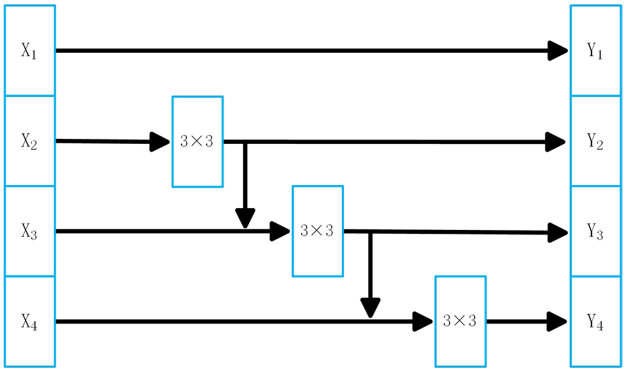


Fig. 3 Res2Net module

The third uses 12 virtual cameras, with the interval angle $d=30$ degrees to obtain 12 views as shown in Fig. 2b. Our method can also be used to generate multi-view from other perspectives.

3.2 Depth extraction visual feature module

3.2.1 Extracting visual features from views using Res2Net

For a set of multi-view $V = \{V_1, V_2, V_3, \cdots, V_n\}$ of 3D models, we use Res2Net [37] to extract visual features, increasing the number of receptive fields, making feature extraction more powerful and reducing information loss during feature extraction. As shown in Fig. 3, the Res2Net module replaces the underlying block in the ResNet structure. First, the convolution layer of 3×3 is evenly divided into p subsets, represented by $x = \{x_1, x_2, x_3, \cdots, x_p\}$. Each subset (excluding x_1) is then input into a 3×3 convolution denoted as $Conv_p$. From x_3 onward, the output of $Conv_p$ is added before the input of $Conv_{p-1}$, increasing the possible perception domains in a single layer. The processing formula of the Res2Net module can be expressed as:

$$y_p = \begin{cases} x_p & \Big| p = 1; \\ Conv_p(x_p) & \Big| p = 2; \\ Conv_p(x_p + y_{p-1}), & \Big| p \geq 3, p \text{ is an integer} \end{cases} \tag{1}$$

where $y = \{y_1, y_2, y_3, \cdots, y_p\}$ is the output of Res2Net module. It is then connected and transferred to a 1×1 convolution layer to ensure the channel size for the residual module of Res2Net.

3.2.2 Attention-convolution pooling

First, we convert the multi-view visual features extracted by Res2Net from the 2D image into a feature map of size $m\times n$, which is represented as input. Our attention-convolution pooling method has two main parts, as shown in Fig. 4. The first part uses the attention mechanism to extract characteristic information from the view; the second part uses the convolution operation to extract detailed information from the view.

In the first stage, for $f_{m\times n}$, we generate three feature maps from four 1×1 convolutions: $Query$, Key_1, Key_2, and $Value$. The process is described by formula (2):

$$\begin{aligned} Query &= Conv_{1\times1}(f_{m\times n}) \\ Key_1 &= Conv_{1\times1}(f_{m\times n}) \\ Key_2 &= Conv_{1\times1}(f_{m\times n}) \\ Value &= Conv_{1\times1}(f_{m\times n}) \end{aligned} \tag{2}$$

where $f_{m\times n}$ represents an input characteristic map of size $m\times n$, $Conv_{1\times1}$ represents the convolution operation using a 1×1 convolution core, and $Query$, Key_1, Key_2 and $Value$ are the characteristic map obtained after the convolution operation using a 1×1 convolution core.

Next, we transpose the eigenvalue $Query$ to the eigenvalue Q^T of size $n\times m$, obtaining the two $n\times n$ eigenvalues $f_{(n\times n)_1}$ and $f_{(n\times n)_2}$ via a product operation with the eigenvalues Key_1 and Key_2, respectively. The process is described by formula (3):

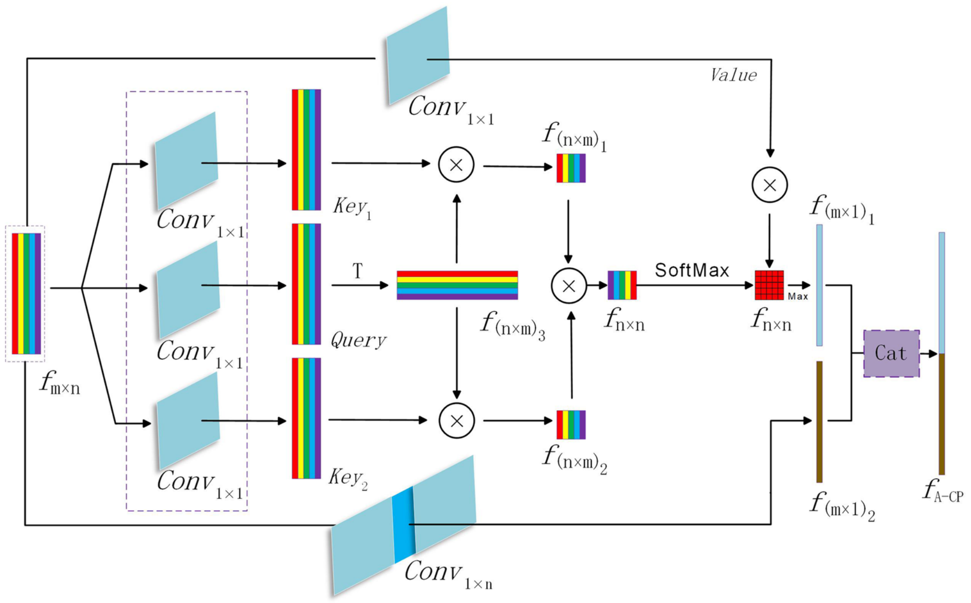


Fig. 4 Our proposed attention-convolution pooling method

$$
\begin{aligned}
&Q^T = Conv_{1\times 1}(f^T_{m\times n})\\
&f_{(n\times n)_1} = Q^T \otimes Key_1\\
&f_{(n\times n)_2} = Q^T \otimes Key_2
\end{aligned}
\tag{3}
$$

where T represents the transpose of the signature map, and $\otimes$ represents the product operation between the two signatures.

The resulting $f_{(n\times n)_1}$ and $f_{(n\times n)_2}$ are multiplied again to obtain an $n\times n$ feature map recorded as $f_{(n\times n)}$. Then, *Value* is multiplied with the attention weight once, with the max pooling used to reduce its dimensionality, producing an m×1 feature vector $f_{(m\times 1)_1}$. The process is described by formula (4):

$$f_{(m\times 1)_1} = Max(Value \otimes soft\max(f_{(n\times n)_1} \otimes f_{(n\times n)_2})) \tag{4}$$

where *softmax* represents the softmax activation function, and *Max* represents the maximum pooling operation.

In the second stage, we generate an $m\times 1$ signature graph from a convolution layer of $1\times n$ with the original signature $f_{m\times n}$, which is denoted as $f_{(m\times 1)_2}$. The process is described by formula (5):

$$f_{(m\times 1)_2} = Conv_{1\times n}(f_{m\times n}) \tag{5}$$

where $f_{m\times n}$ represents the input characteristic map of $m\times n$ size, $f_{(m\times 1)_2}$ represents the characteristic map of convolution operation with a $1\times n$ convolution kernel, and $Conv_{1\mathrm{x}n}$ is the convolution operation with the $1\times n$ convolution kernel.

After completing the first and second parts, we stitch two $m\times 1$ eigenvectors $f_{(m\times 1)_1}$ and $f_{(m\times 1)_2}$ into 2 $m\times 1$ eigenvectors described by formula (6):

$$f_{A-CP} = \mathrm{Cat}(f_{(m\times 1)_1}, f_{(m\times 1)_2}) \tag{6}$$

where *Cat* denotes the splicing operation, and f_{A-CP} is the eigenvector obtained after the attention-convolution pooling.

After sorting it out, the results are as follows:

$$
\begin{aligned}
f_{A-CP} = \mathrm{Cat}(&Max(Value \otimes soft\max((Conv_{1\times 1}(f_{m\times n})\\
&\otimes Conv_{1\times 1}(f_{m\times n})) \otimes (Conv_{1\times 1}(f^T_{m\times n})\\
&\otimes Conv_{1\times 1}(f_{m\times n})))) + Conv_{1\times n}(f_{m\times n}))
\end{aligned}
\tag{7}
$$

3.3 Multi-view visual feature fusion module

In this section, we present our solution to the multi-view visual feature fusion classification problem. From the previous operations, we obtain a feature vector 2 $m\times 1$ that represents the feature and detail information for each view. Here we add a fully connected layer to produce a $C\times 1$ eigenvector by using the softmax function to solve the classification problem. It then obtains the probability distribution of the model to be classified, as shown in Fig. 5. The process is described by formula (8).

$$\hat{y} = softmax(y) = softmax(wx + b) \tag{8}$$

where x is the input of the full connection layer, w is the weight, b is the offset, and $\hat{y}$ is the output softmax probability. Softmax is calculated as

$$softmax(y_c) = \frac{e^{y_c}}{\sum_{c=1}^{C} e^{y_c}} \tag{9}$$

where C is the dataset category. For example, we set C to 40 for use with the ModelNet40 dataset.

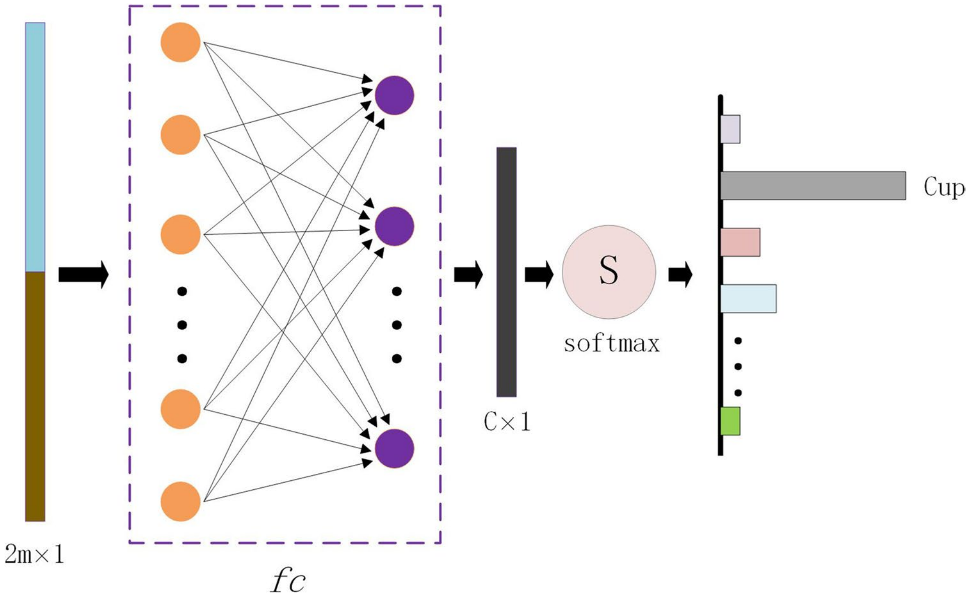


Fig. 5 Visual feature fusion classification module

4 Experiment

4.1 Dataset

To assess the performance of our proposed method, we performed experiments on a standard 3D CAD model dataset, ModelNet40 [44], with related classification tasks and compared them to related methods. ModelNet40 has 12,311 models, divided into 9843 training models and 2468 test models, representing 40 common CAD model categories.

4.2 Evaluation metrics

Because the number of models is not the same each class, we measure the classification performance of our method using the overall accuracy OA [45] for each sample and the average accuracy AA [46] for each category. These values are defined as follows.

Overall accuracy (OA): The ratio of the number of samples correctly classified to the total number of samples, expressed as

$$OA=\frac{1}{N}\sum_{i=1}^{C}x_{ii} \tag{10}$$

where N is the total number of samples, x_{ii} is the number of correct classifications distributed diagonally along the confusion matrix, and C is the category of the dataset.

Average accuracy (AA) for each category: The average of the ratio of the number of correct predictions for each category to the total number of predictions each category, expressed as

$$AA=\frac{\text{sum(recall)}}{C} \tag{11}$$

where *recall* represents the ratio of predicted pairs to actual samples, and C represents the number of categories.

4.3 Experiment setup and analysis

The computer used for the experiment was equipped with two NVidia Titan Xp GPUs and 64 GB of memory. We used PyTorch [47] for all of the experiments. In the experiment, we configured the two training stages to 10 and 20 iterations, respectively. In the first stage, we classify only a single picture for fine-tuning the model. In the second stage, we train all views of the voxelized model of the original 3D point cloud model to train the entire classification framework. In the test phase of the experiment, we only tested the second phase.

To optimize the overall architecture, we used Adam [48] as the two-stage optimizer. In addition, we set the learning rate decay and weight decay, with the initial learning rate (lr) value set to 0.0001, after which we adjusted the next learning rate to half of the previous one. The weight decay uses L2 regularization to speed up the training of the model and reduce over-fitting.

For extracting visual features of the views, we compared the VGG-11 model proposed by Simonyan [49], the ResNet-50 model proposed by He [36], the Res2Net-50 and Res2NeXt-50 model proposed by Gao [37], and the DenseNet-121 model proposed by Huang [50] as the backbone model of depth extraction visual features module in our framework. The results are shown in Table 1. Here, the learning rate (lr) was set to 5×10^{-5}, with the batch sizes

Table 1 The effect of different backbone models on classification performance

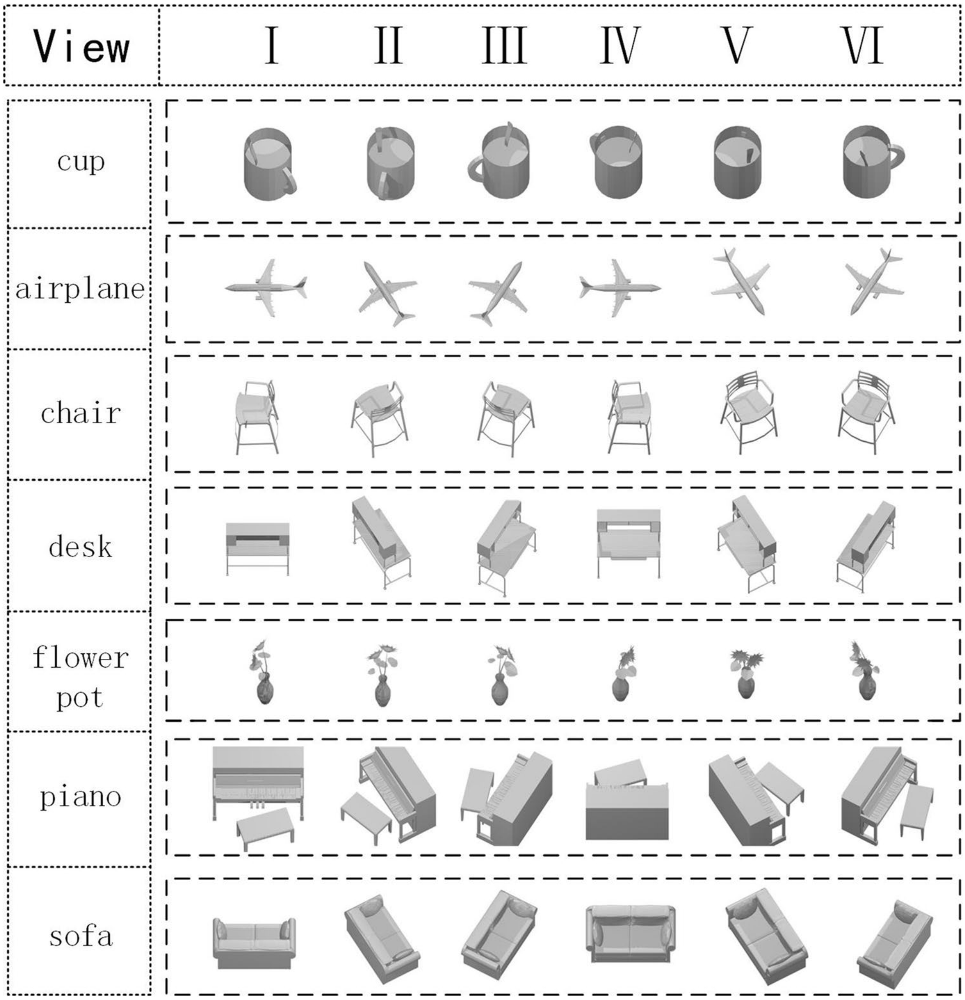
Network	ModelNet40 (%)	
	OA	AA
VGG-11[49]	91.06	88.32
ResNet-50[37]	91.79	89.29
Res2Net-50[36]	92.20	90.24
Res2NeXt-50[36]	92.37	90.39
DenseNet-121[50]	91.32	88.85

for the first (bs_1) and second stages (bs_2) set to 64 and 16, respectively. For the feature pooling module, we used the most classic pooling method, MaxPooling, with N set to 6. Both Res2Net-50 and Res2NeXt-50 both exceeded 92% and 90% in OA and AA performance, so we selected these two models as the backbone models for our subsequent experiments.

4.4 The influence of different numbers of views on classification performance

Before discussing the effect of different N values on classification performance in N perspective, we first make a comparison between multiple and single views. Six views of a cup, as shown in Fig. 6, contain different feature information. View V ignores the key feature information of the handle, and the handle information of view IV is not obvious. When feature extraction is performed on 2D view images, the loss of feature information is bound to affect classification accuracy if single view is used for experimentation, and the single view is cup view IV or view V. In addition, multiple views of other objects are given. From Fig. 6, we can see that each view of some 3D objects (e.g., airplane and chair) has distinct characteristics, while some 3D objects (e.g., desk and piano) have similar view characteristics. If feature extraction is carried out in a single view, it is easy to confuse the feature information of objects, resulting in classification errors. Multi-view can fuse the feature information under each view, which can make up for the shortage

Fig. 6 Six views of different objects



of feature information easily lost by single view. However, the ensuing problem is determining how many views will achieve the best classification accuracy. For this reason, we will further explore the best value of N in N perspectives.

When discussing the effect of different number of perspectives N on classification performance, we conducted experiments with N set to 3, 6, and 12. The experimental results are shown in Table 2. Adjusting the hyperparameters of learning rate and batch size can improve performance, so we set $lr = 1 \times 10^{-4}$, $bs_1 = 128$, and $bs_2 = 32$.

Our experimental comparison shows that our model framework outperformed other methods (such as MVCNN [28], RCPCNN [51] and GVCNN [29]). In both 6 and 12 perspectives, our approach achieves a better level of OA performance by more than 93%. Our method performed best with $N = 6$ perspectives. Moreover, increasing the number of perspectives N also increased the training time. Within our framework, the classification performance is compared between Res2Net-50 and Res2NeXt-50 as a backbone model. The OA and AA of Res2NeXt-50 with 6 perspectives were 93.64% and 91.53%, respectively. Thus, the backbone model Res2NeXt-50 and N = 6 are selected as optimal configuration in our MVACPN.

4.5 Experimental results and analysis

In examining our results, we consider both the classification performance and optimal configuration. We compared our methods with other advanced methods, including those based on voxels [16–18], point clouds [19–21, 25], and views [28, 29, 32]. The comparison results are shown in Table 3.

Table 2 The influence of different numbers of perspectives on classification performance

Methods	N-view	ModelNet40 (%)	
		OA	AA
MVCNN [28]	3	91.30	—
	6	92.00	—
	12	92.10	89.90
RCPCNN [51]	3	92.10	—
	6	92.20	—
	12	92.20	—
GVCNN [29]	12	92.60	—
Ours (Res2Net-50)	3	92.25	90.06
	6	**93.40**	90.85
	12	93.26	**91.03**
Ours (Res2NeXt-50)	3	92.12	90.29
	6	**93.64**	**91.53**
	12	93.27	91.16

Table 3 Comparison of classification results with the ModelNet40 dataset

Methods	Modality	OA (%)	AA (%)
O-CNN [17]	Voxel	90.60	–
VoxNet [16]	Voxel	–	83.00
VRN [18]	Voxel	91.33	–
PointNet [19]	Point Cloud	89.20	86.00
PointNet++ [20]	Point Cloud	91.90	–
KD-Net [21]	Point Cloud	91.80	88.50
PointGrid [25]	Point Cloud	92.00	88.90
MVCNN [28]	12-Views	92.10	89.90
GVCNN [29]	8-Views	93.10	–
MHBN [32]	12-Views	93.40	–
MVACPN (Ours)	6-Views	**93.64**	**91.53**

The results show that, compared with other methods, our proposed algorithm framework outperformed the other methods, with 93.64% and 91.53% classification accuracy scores in terms of OA and AA. Especially on AA, our method is significantly higher than others. It is worth noting that compared with other multi-view methods (including MVCNN and MHBN with 12 views and GVCNN with 8 views), our method, MVACPN, achieves the best classification accuracy with only 6 views. In addition, a reduced view can also shorten the training time. At the same times, we show a confusion matrix visualization of different object classifications on the ModelNet40 dataset by MVACPN, as shown in Fig. 7, where the values on the diagonal line of the obfuscation matrix represent the correct number of classifications and those outside the diagonal line represent the number of classification errors. It is worth noting that it is not the greater the number on the diagonal that is better, but the smaller the number outside the diagonal, the better. It can be seen that the values of the confusion matrix are mainly concentrated on the diagonal lines, even reaching 100% in the values of airplane, car, and so on, which indicates that our method has a very good classification effect.

We credit the higher performance to two main factors. The first is that MVACPN further increases the number of acceptable domains by introducing Res2Net, making feature extraction more powerful and reducing the loss of information during feature extraction. The second is that the attention-convolution feature pooling module in MVACPN includes both attention and convolution operations to make full use of the attention mechanism when extracting feature information from views and the convolution operation for details. Compared with traditional methods, this method finds more useful information related to the current output in the input data, effectively solving the loss of feature information caused by feature representation and view details

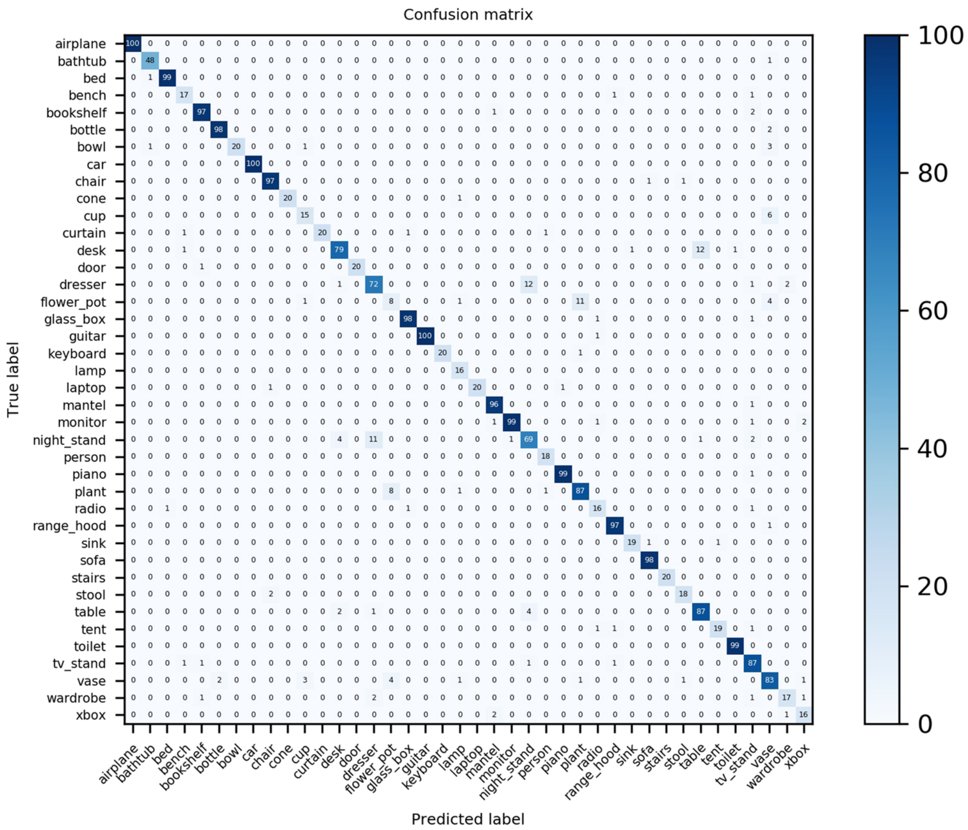


Fig. 7 Confusion matrix visualization of different object classifications for MVACPN on ModelNet40 dataset

caused by dimension reduction, thus improving the accuracy of classification.

4.6 Ablation study

To better illustrate the performance of MVACPN, we added a set of ablation experiments to explore the performance differences of different CNN frameworks in MVACPN, where we assigned a viewing angle of 6. Ablation experiments are shown in Table 4, where MaxP represents MaxPooling and ACP represents attention-convolution pooling in our proposed MVACPN framework.

Therefore, we can see that the combination of VGG, ResNet, Res2Net, and Res2NeXt with ACP improves OA and AA relatively compared with the combination of MaxPooling and MVACPN framework. This demonstrates the validity of the ACP we have proposed. At the same time, DenseNet leads to a decrease in OA and AA in MVACPN, possibly due to the fitting of DenseNet's dense connection structure to our ACP structure. From this ablation study, we can see that the proposed ACP, combined with Res2NeXt to form MVACPN framework, achieves the best performance in 3D point cloud classification tasks.

5 Conclusions

In this study, we propose a multi-view attention-convolution pooling network framework (MVACPN) for high-precision classification of 3D point clouds. Considering the loss of feature information caused by feature representation and the loss of detail information in views during dimension reduction, we propose an attention-convolution pooling structure,

Table 4 OA and AA in ablation experiments

Methods	ModelNet40 (%)	
	OA	AA
VGG-11 [43] + MaxP	91.06	88.32
VGG-11 [43] + ACP	91.97	88.93
ResNet-50 [37] + MaxP	91.79	89.29
ResNet-50 [37] + ACP	92.01	89.93
Res2Net-50 [36] + MaxP	92.20	90.24
Res2Net-50 [36] + ACP	93.40	90.85
DenseNet-121 [44] + MaxP	91.32	88.85
DenseNet-121 [44] + ACP	90.94	88.61
Res2NeXt-50 [36] + MaxP	92.37	90.39
Res2NeXt-50 [36] + ACP(Ours)	93.64	91.53

which can be more focused on finding useful information related to the current output in the input data for processing and effectively resolving the loss of feature information caused by the feature representation, as well as the loss of detail information for each view in the dimension reduction process, so as to improve the accuracy of classification. We ran multiple experiments to obtain optimal configuration settings to achieve the best classification accuracy with the ModelNet40 dataset. The results show that our framework achieves higher classification accuracy compared with other contemporary methods. Moreover, the algorithm can also be applied in other domains like intelligent packaging technology to demonstrate its generality.

Acknowledgements The financial support for this work was sponsored by Natural Science Foundation of Shanghai under Grant 19ZR1435900.

References

1. Chiang CH, Kuo CH, Lin CC et al (2020) 3D point cloud classification for autonomous driving via dense-residual fusion network. IEEE Access 8:163775–163783
2. Yang L, Liu Y, Peng J et al (2020) A novel system for off-line 3D seam extraction and path planning based on point cloud segmentation for arc welding robot. Robot Comput Integr Manuf 64:101929
3. Li X, Du S, Li G et al (2020) Integrate point-cloud segmentation with 3D LiDAR scan-matching for mobile robot localization and mapping. Sensors 20(1):237
4. Bolkas D, Chiampi J, Chapman J et al (2020) Creating a virtual reality environment with a fusion of sUAS and TLS point-clouds. Int J Image Data Fusion 11(2):136–161
5. Yao L, Jiang P, Xue Z et al (2020) Graph convolutional network based point cloud for head and neck vessel labeling. In: International Workshop on Machine Learning in Medical Imaging. Springer, Cham, pp 474–483
6. Yang L, Chakraborty R (2020) A GMM based algorithm to generate point-cloud and its application to neuroimaging. In: 2020 IEEE 17th International Symposium on Biomedical Imaging Workshops (ISBI Workshops). IEEE, pp 1–4
7. Mercado-Ravell DA, Castillo P, Lozano R (2019) Visual detection and tracking with UAVs, following a mobile object. Adv Robot 33(7–8):388–402
8. Yang X, Wang H, Chen S et al (2019) Cascaded network with deep intensity manipulation for scene understanding. Comput Anim Virtual Worlds 30(3–4):e1888
9. Kaesemodelpontes J et al (2017) Compact model representation for 3D reconstruction. In: 7th IEEE International Conference on 3D Vision, 3DV2017, 29
10. Kim MK, Thedja JPP, Chi HL et al (2021) Automated rebar diameter classification using point cloud data based machine learning. Autom Constr 122:103476
11. Chen J, Wang Z, Chen J et al (2019) Design and research on intelligent teaching system based on deep learning. Comput Sci 6:550–554
12. Yang S, Xu M, Xie H et al (2021) Single-view 3D object reconstruction from shape priors in memory. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3152–3161
13. Ye H, Du Z, Cao F (2021) A novel 3D shape classification algorithm: point-to-vector capsule network. Neural Comput Appl. https://doi.org/10.1007/s00521-021-06231-z
14. Zou W, Wu D, Tian S et al (2021) End-to-end 6DoF pose estimation from monocular RGB images. IEEE Trans Consum Electron 67(1):87–96
15. Gao Z, Li Y, Wan S (2020) Exploring deep learning for view-based 3D model retrieval. ACM Trans Multimed Comput Commun Appl 16(1):1–21
16. Maturana D, Scherer S (2015) Voxnet: a 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, pp 922–928
17. Wang PS, Liu Y, Guo YX et al (2017) O-cnn: octree-based convolutional neural networks for 3d shape analysis. ACM Trans Graph 36(4):1–11
18. Brock A, Lim T, Ritchie JM et al (2016) Generative and discriminative voxel modeling with convolutional neural networks. arXiv preprint. http://arxiv.org/abs/1608.04236. Accessed 25 Feb 2021
19. Qi CR, Su H, Mo K et al (2017) Pointnet: deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 652–660
20. Qi CR, Yi L, Su H et al (2017) Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in neural information processing systems, pp 5099–5108
21. Klokov R, Lempitsky V. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models[C]//Proceedings of the IEEE International Conference on Computer Vision. 2017: 863–872.
22. Riegler G, Osman Ulusoy A, Geiger A (2017) Octnet: Learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3577–3586
23. Li Y, Bu R, Sun M et al (2018) Pointcnn: Convolution on x-transformed points. In: Advances in neural information processing systems, pp 820–830
24. Wang Y, Sun Y, Liu Z et al (2019) Dynamic graph cnn for learning on point clouds. Acm Trans. Graph. 38(5):1–12
25. Le T, Pointgrid DY (2018) A deep network for 3d shape understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 9204–9214
26. Wang F, Hu H, Ge X et al (2020) Multientity registration of point clouds for dynamic objects on complex floating platform using object silhouettes. IEEE Trans Geosci Remote Sens 59(1):769–783
27. Gao Y, Tang J, Hong R et al (2011) Camera constraint-free view-based 3-D object retrieval. IEEE Trans Image Process 21(4):2269–2281
28. Su H, Maji S, Kalogerakis E et al (2015) Multi-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE international conference on computer vision, pp 945–953

29. Feng Y, Zhang Z, Zhao X et al (2018) Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 264–272
30. Jiang J, Bao D, Chen Z et al (2019) MLVCNN: multi-loop-view convolutional neural network for 3D shape retrieval. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 8513–8520
31. Nie W, Liang Q, Liu AA et al (2019) MMJN: multi-modal joint networks for 3D shape recognition. In: Proceedings of the 27th ACM International Conference on Multimedia, pp 908–916
32. Yu T, Meng J, Yuan J (2018) Multi-view harmonized bilinear network for 3d object recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 186–194
33. Sun X, Lian Z, Xiao J (2019) SRINet: Learning Strictly RotationInvariant Representations for Point Cloud Classification and Segmentation. In: Proceedings of the 27th ACM International Conference on Multimedia, pp 980–988
34. Zhou Y, Zeng F, Qian J et al (2019) 3D shape classification and retrieval based on polar view. Inf Sci 474:205220
35. Chao H, He Y, Zhang J et al (2019) Gaitset: Regarding gait as a set for cross-view gait recognition. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol 33, pp 8126–8133
36. Gao S, Cheng M M, Zhao K et al (2019) Res2net: A new multi-scale backbone architecture. IEEE transactions on pattern analysis and machine intelligence
37. He K, Zhang X, Ren S et al (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
38. Wang H, Yang Y, Liu B et al (2019) A study of graph-based system for multi-view clustering. Knowl-Based Syst 163:1009–1019
39. Xiao Q, Dai J, Luo J et al (2019) Multi-view manifold regularized learning-based method for prioritizing candidate disease miRNAs. Knowl-Based Syst 175:118–129
40. Zhang Y, Yang Y, Li T et al (2019) A multitask multiview clustering algorithm in heterogeneous situations based on LLE and LE. Knowl-Based Syst 163:776–786
41. Zhang X, Yang Y, Li T et al (1895) CMC: a consensus multi-view clustering model for predicting Alzheimer's disease progression. Comput Methods Programs Biomed 2021:105895
42. Hayashi T, Fujita H, Hernandez-Matamoros A (2021) Less complexity one-class classification approach using construction error of convolutional image transformation network. Inf Sci 560:217–234
43. Wu Y, Jiang X, Fang Z et al (2021) Multi-modal 3D object detection by 2D-guided precision anchor proposal and multi-layer fusion. Appl Soft Comput 108:107405
44. Wu Z, Song S, Khosla A et al (2015) 3d shapenets: a deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1912–1920
45. Uy MA, Pham QH, Hua BS et al (2019) Revisiting point cloud classification: a new benchmark dataset and classification model on real-world data. In: Proceedings of the IEEE International Conference on Computer Vision, pp 1588–1597
46. Zhai R, Li X, Wang Z et al (2020) Point cloud classification model based on a dual-input deep network framework. IEEE Access 8:55991–55999
47. Paszke A, Gross S, Chintala S et al (2017) Automatic differentiation in pytorch
48. Zhang Z (2018) Improved adam optimizer for deep neural networks. In: 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). IEEE, pp 1–2
49. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint. http://arxiv.org/abs/1409.1556. Accessed 24 Feb 2021
50. Huang G, Liu Z, Van Der Maaten L et al (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
51. Wang C, Pelillo M, Siddiqi K (2019) Dominant set clustering and pooling for multi-view 3d object recognition. arXiv preprint. http://arxiv.org/abs/1906.01592. Accessed 22 Feb 2021

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Wenju Wang born on March 27, 1979, China. Current position: lecturer at University of Shanghai for Science and Technology, China. He received PHD degree in computer application technology from Tongji University, China, in 2012. His research interests include computer 3D vision, 3D shape recognition, and deep learning.

Tao Wang is currently pursuing the M.S. degree at University of Shanghai for Science and Technology. His research interests include computer vision, 3D point cloud classification and segmentation, and deep learning.

Yu Cai is currently pursuing the M.S. degree at University of Shanghai for Science and Technology. His research interests include computer vision, 3D point cloud classification, 3D shape recognition, and deep learning.