



Multi-view dual attention network for 3D object recognition

Wenju Wang¹ · Yu Cai¹ · Tao Wang¹

Received: 6 February 2021 / Accepted: 24 September 2021 / Published online: 14 October 2021
© The Author(s) 2021

Abstract

The existing view-based 3D object classification and recognition methods ignore the inherent hierarchical correlation and distinguishability of views, making it difficult to further improve the classification accuracy. In order to solve this problem, this paper proposes an end-to-end multi-view dual attention network framework for high-precision recognition of 3D objects. On one hand, we obtain three feature layers of query, key, and value through the convolution layer. The spatial attention matrix is generated by the key-value pairs of query and key, and each feature in the value of the original feature space branch is assigned different importance, which clearly captures the prominent detail features in the view, generates the view space shape descriptor, and focuses on the detail part of the view with the feature of category discrimination. On the other hand, a channel attention vector is obtained by compressing the channel information in different views, and the attention weight of each view feature is scaled to find the correlation between the target views and focus on the view with important features in all views. Integrating the two feature descriptors together to generate global shape descriptors of the 3D model, which has a stronger response to the distinguishing features of the object model and can be used for high-precision 3D object recognition. The proposed method achieves an overall accuracy of 96.6% and an average accuracy of 95.5% on the open-source ModelNet40 dataset, compiled by Princeton University when using Resnet50 as the basic CNN model. Compared with the existing deep learning methods, the experimental results demonstrate that the proposed method achieves state-of-the-art performance in the 3D object classification accuracy.

Keywords Object recognition · Multi-view · Deep learning · Computer vision

1 Introduction

The rapid development of 3D sensing technology has led to the development of depth cameras, laser scanners, depth scanners, and other 3D cameras and scanning equipment. 3D data acquisition is becoming increasingly convenient and accurate, promoting the continuous expansion of its application fields and scenes. Compared with multi-cameras, 3D sensor imaging devices such as depth cameras can capture a large amount of detailed 3D object structure information directly and conveniently [1]. Therefore, depth sensors have been widely used in autonomous driving [2], robots [3], augmented reality [4], reverse engineering [5],

and medicine [6]; however, they also face many problems. 3D object recognition is one of the most urgent problems in the above application fields and has become the current research hotspot. 3D object recognition research is divided into two categories according to different methods: early traditional methods and recent deep learning methods, and traditional methods have given way to those based on deep learning. Early methods depended on artificial 3D features and machine learning, while recent deep learning methods include voxel-based, pointset-based and view-based approaches.

View-based methods have achieved the best performance so far. Compared to other input methods, they are relatively low-dimensional, independent of complex 3D features, and robust to representation of 3D objects. They easily capture input views and are not limited to 3D data. They benefit from mature CNN models such as VGG [7], GoogLeNet [8], ResNet [9], and DenseNet [10]. These mature models enable the learning of view features and enhanced view representation. How to aggregate the

✉ Yu Cai
192402566@st.usst.edu.cn

¹ College of Communication and Art Design, University of Shanghai for Science and Technology, Shanghai 200093, China

resultant multi-view features to form differentiated shape descriptors is important to optimal 3D shape recognition. MVCNN [11] uses view-pooling to aggregate features from view descriptors, which provides direction for multi-view 3D object recognition. However, MVCNN [11] treats all views equally, and the use of max pooling to preserve the largest elements in a view can result in a loss of information, ignoring content relationships of views and distinguishability between them, which greatly limits the performance of view shape descriptors. On one hand, some view contents are similar, and they contribute similarly to shape descriptors without highlighting the key information for distinguishability. On the other hand, different views may not be effectively related, and the relative location information between views is ignored.

Information related to different perspectives of objects is especially important in human visual perception. Hence, it is important to study intrinsic correlation and discover distinct features.

2 Related work

2.1 Handcrafted descriptors

Early 3D shape recognition was mainly based on artificially designed 3D data description features and machine learning methods. Ozbay et al. [12] used a fine Gaussian support vector machine for 3D object recognition and a Zernike moment (ZM) for 3D feature extraction. Li et al. [13] extracted geometric, color, and intensity features of 3D shapes obtained by ground laser scanning (TLS) based on super pixel neighborhoods and used random forest classification. This method can only be used to distinguish vegetation from curb stones, but its applicability is not strong. Chen et al. [14] proposed a hybrid kernel support vector machine 3D point cloud classification algorithm based on the combined features of normalized elevation, elevation standard deviation, and elevation difference of 3D point cloud data. The method needs improvement in feature extraction optimization. Based on artificial design, traditional methods tend to exhibit accuracy problems and insufficient robustness.

2.2 Voxel-based approaches

Voxel-based approaches learn 3D features from voxels, which represent 3D shapes through the distribution of corresponding binary variables. Early deep learning methods usually used a 3D convolutional neural network (CNN) to build on voxel representation. Maturana et al. [15] proposed VoxNet, a volume-occupying network, to achieve robust 3D target recognition. VoxNet converts 3D

data to regular 3D voxel data and classifies it based on its spatial local correlation. The voxel structure is limited by its resolution due to sparse data. Wu et al. [16] proposed ShapeNets. Based on VoxNet, it represents 3D shapes as probability distributions of binary variables on a voxel grid, which learns the distributions of points from a variety of 3D shapes. Riegler et al. [17] proposed OctNet, which hierarchically divides point clouds using a structure that represents several shallow octrees along a regular grid in a scene. The structure is coded using bit string representation, and the eigenvectors of each voxel are indexed by a simple algorithm. Le et al. [18] proposed a hybrid point grid network that integrates point and grid representations. Sampling a fixed number of points within each embedded volume grid cell allows the network to extract geometric details using 3D convolution. The computational and memory footprint is not well extended to dense 3D data due to the stereoscopic increase in resolution. The accuracy of data in voxel-based methods depends on its resolution, which limits its development due to the amount of computation.

2.3 Pointset-based approaches

Point cloud is another type of 3D data structure that can be understood as sampling points on the surface of 3D objects. These points are unevenly distributed, and each consists of 3D position information in 3D space. Charles Qi et al. proposed PointNet [19] and PointNet++ [20]. PointNet [19] uses multiple MLP layers to learn point cloud features and a max pooling layer to extract global shape features. Multiple MLP layers are used to obtain the classification score, and nonlinear transformation achieves permutation invariance. They designed an infrastructure for various applications, including shape classification and segmentation. Since the features of each point in PointNet [19] are independently learned, it is impossible to capture the local structure information between points. PointNet++ [20] solves this problem through local information extraction so as to realize better classification. PointNet++ [20] includes sampling, grouping, and PointNet layers. It adds multiple levels of abstraction to learn features from local geometry and abstract local features, layer by layer. Many subsequent frameworks have exploited the simplicity and strong expression ability of PointNet [19]. Achlioptas et al. [21] introduced a deep automatic encoder network to learn point cloud representation. It uses five 1D convolutional layers, ReLU nonlinear activation, batch normalization, and max pooling layers to learn independent learning point features. Mo-Net [22] has a structure similar to that of PointNet [19], but it needs a finite set of moments as input. Pointweb [23] is based on PointNet++, and it uses the context of local neighborhoods to use adaptive feature adjustment (AFA) to

improve point features. Lin et al. [24] accelerated the reasoning process by constructing a lookup table for the input and function spaces of PointNet learning. This method of directly using unordered point cloud as input has always been the pursuit of 3D object recognition. Its prominent problem is the lack of high-quality training datasets. Although there are many related 3D datasets, they are still not comparable to the size of 2D image datasets. Another challenge of the original point cloud method is the disorder and non-uniformity of point cloud data, which hinders the direct application of spatial convolution.

2.4 View-based approaches

View-based approaches render from multiple angles of a 3D object. The multi-view method renders 3D objects into multiple 2D views, extracts the corresponding view features, and fuses them for accurate 3D object recognition. The aggregation of multiple view features into a distinct global representation is a key challenge. Su et al. [11] proposed MVCNN, a standard CNN structure trained to independently recognize shape rendering views. Recognition increases when multiple locations of views are provided. MVCNN aggregates information from multiple views of a 3D model into a global descriptor. However, maximum pooling retains only the largest elements in a view, resulting in a loss of information. To solve this problem, Yu et al. [25] proposed MHBN, which integrates local convolution features by coordinating bilinear pools on the basis of MVCNN, resulting in a compact global descriptor to extract global information. Wang et al. [26] introduced a view clustering and pooling layer based on dominance sets. They looped together the features from each set as input to the same layer. Recently, Xie [27] proposed a new multi-view network graph embedding method to aggregate view node representations by integrating node information from multiple views with attention, and this attention method has attracted much research attention. View-based 3D object recognition can use a large number of datasets such as ImageNet [28] for pre-training, and can directly use the rendered 2D perspective image on the 2D CNN to achieve class-level recognition performance of more than 93%. The accuracy of view-based 3D object recognition and classification has great room for improvement. The main methods are wavelet transform [29] and attention mechanism [30–32]. Wavelet transform is used to denoise in preprocessing of the collected view images, which aims to improve the classification accuracy by improving the quality of the view images. Attention mechanism refers to the human visual attention thinking mode. With a limited number of views, it can filter out the target areas of important value information from a large number of unrelated background areas. It can process

visual information efficiently and help to improve the recognition and classification accuracy, which deserves further study.

2.5 Attention modules

The attention models (AM) have performed well in different applications of neural networks. Attention mechanisms can be explained in terms of the human visual mechanism. Human vision quickly scans a global image to obtain a focus on the target area, known as the focus of attention, and concentrates on this area to obtain more detailed information on the target, while suppressing useless information. Many studies have combined attention modules in neural networks to perform various tasks. The attention models can be roughly divided into spatial domain attention, channel attention, and hybrid domain models. Spatial area attention can be understood as the area where a neural network is looking. Jaderberg et al. [30] proposed spatial transformer networks (STN), which adaptively and spatially transform and align input data based on classification or other visual tasks to preserve key information. Channel attention can be understood as what a neural network is looking at. There are many convolution cores in each layer of a CNN, each corresponding to a characteristic channel. In contrast to spatial attention, channel attention allocates resources between convolution channels. Hu et al. [31] proposed the squeeze-and-exchange network (SENet) to improve the quality of representation of network generation by explicitly modeling the interdependence between channels with their convolution characteristics. The hybrid domain model allows a neural network to capture richer information by combining the attention of both spatial and channel domains. Woo et al. [32] proposed that convolutional block attention module (CBAM) uses the channel and spatial attention modules in turn to emphasize meaningful features in both spatial and channel dimensions. The attention mechanism can invest more attention resources in the target area of the focus and gather more detailed information about the target, which is exactly what we need for view-based approaches.

3 Contributions

Based on the above analysis, we propose a multi-view dual attention network (MVDAN), as shown in Fig. 1, based on a view space attention block (VSAB) and view channel attention block (VCAB). VSAB explores relationships between regions within a view to enhance its distinctive characteristics. It generates a spatial attention weight matrix that adaptively collects global context information to enhance the responsiveness of the more discriminatory

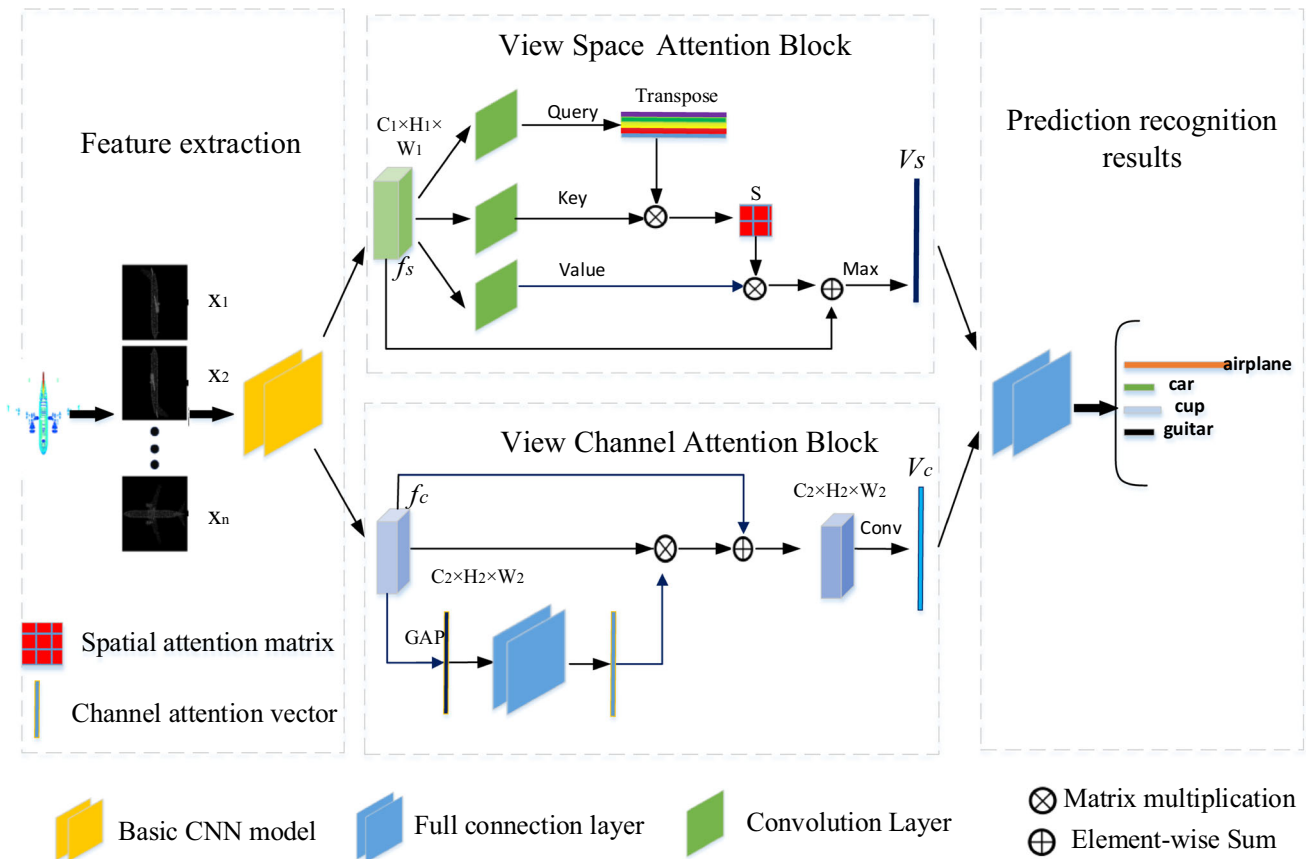


Fig. 1 The architecture of multi-view dual attention network (MVBAN). A 3D shape is rendered in different directions, and the features of the views are extracted from the rendered image via the basic CNN model. These views are then grouped together by two attention blocks to obtain a compact shape descriptor passed into the full connection layer to complete the classification

details within a view. VCAB studies the correlation between views, generates channel attention weight vectors, scales weights to focus on important perspectives, and enhances the relative discrimination of views. It examines view-to-view relationships to learn the relative importance of each view.

An experiment on the ModelNet40 dataset demonstrates the effectiveness of the proposed method, which uses rendered multi-view input for advanced performance. Our main contributions can be summarized as follows.

1. We propose MVDAN, which uses VSAB and VCAB to generate a global shape descriptor for high-precision 3D object recognition.
2. VSAB obtains the spatial shape descriptor of views, which emphasizes the importance of details with more class discrimination features.
3. VCAB obtains the shape descriptor of a view channel to find the correlation between target perspectives, so as to focus on perspectives with key characteristics.
4. Network MVDAN based on Princeton University's open-source ModelNet40 dataset achieved state-of-the-

art performance compared with existing deep learning-based methods.

The rest of this paper is organized as follows. We introduce the MVDAN architecture in Sect. 4, discuss experiments and their results in Sect. 5, and summarize this work in Sect. 6.

4 Methods

We design a multi-view 3D object recognition network, MVDAN, whose architecture is shown in Fig. 1. It has three parts.

1. The first, a basic CNN model, is for feature extraction of views. The original 3D object model M will be projected from multiple angles to a two-dimensional plane and rendered into n views as input, and feature extraction is performed through the basic CNN model. Each image is fed into the basic CNN model with shared weights. Since the methods of 3D object

recognition based on multi-view [11, 25, 26] adopt VGG-M [7] as the basic CNN model, in order to make a fair comparison, we also introduce VGG-11, which is configured as A in VGG-M [7]. Then, we remove the last full connection layers of the model to access our double attention block.

2. Next comes view feature pooling. As perspectives involve different object directions and structure information, VSAB and VCAB explore relationships within and between views to generate global shape descriptors, respectively.
3. The third part is composed of full connection layers, which input the shape descriptors from pooled features into the fully connected network to complete object recognition. In the next section, we will introduce in detail the multi-view representation and present our proposed view space attention block and view channel attention block.

4.1 Multi-view representations

There are three main data forms of 3D objects: voxels, point clouds, and meshes. This paper deals with the mesh form. 3D objects in the mesh form mainly comprise triangular patches. These triangular patches cover the surface of the object without gaps, forming a hollow 3D model. To produce a multi-view rendering of a 3D shape, we used the Phong [33] reflection model to render a 3D model under a perspective projection, determining the pixel color by interpolating the reflective intensity of the polygon vertices. The projected two-dimensional images are grayscale images, which can reflect the edge information of the object. In order to create multi-view shape representations, we need to set the viewpoints to render each mesh. As set by MVCNN [11] in the first experimental camera, we assume that the input 3D shape is placed vertically on a constant axis (e.g., z axis), and that 12 virtual cameras are placed at 30-degree intervals to render the 3D model with a virtual camera pointing at the center of mass around the 3D model. For comparison purposes, we set up perspectives at every 120° and every 60°. Most models in modern online repositories satisfy the requirement of aligning along a consistent axis, such as 3D Warehouse, and some previous view-based recognition methods follow the same assumption [11, 25, 26]. The elevation of the cameras is 30° from the ground plane and points toward the centroid of the mesh. The centroid is calculated as the weighted average of the centers of the mesh surface, where the weights are the areas of the surface. Finally, each model generates a set of 12 view images. The difference with MVCNN [11] is that we render the images with a black background and set the camera's field of view so that the object is bounded by the

image canvas and rendered as a grayscale image of size 224×224 . We studied the influences of three views (every 120°) and six views (every 60°) in Sect. 5.4.

Based on the above-mentioned multi-view representations, the features of the views are then extracted via the basic CNN model, as shown in Fig. 1.

4.2 View space attention block

After 2D images rendered from different perspectives have learned the view characteristics through the CNN, a distinctive global descriptor is required. Figure 2 shows an example of three perspectives (Views I, II, and III) in the mantel and piano categories of the original dataset rendered by the 3D object. VSAB focuses on some detail for each category (Fig. 3). For example, there are no key features of the mantel category in View I, which makes it difficult to distinguish from the piano. In View II and View III, the keys of the mantel separator and piano become key features to distinguish the categories of the two samples. VSAB makes the more discriminatory details (mantel partition, piano keys) in categories more responsive. Figure 3 shows the structure of VSAB. The workflow can be divided into three phases. First, we get the query, key, and value feature layers by convolutional layer. The first phase generates a spatial attention matrix from key-value pairs that captures the discriminatory details within a view by exploring the spatial relationship between any two pixels of its characteristics. In the second stage, we assign importance to each feature within the value of the spatial branch of the original feature through the spatial attention matrix and add the original input to generate the spatial attention descriptor. In the third stage, we used the max pooling function to aggregate view features to obtain a compact spatial shape descriptor.

Each 3D model M is rendered as n 2D images, which are marked as $X \rightarrow (x_1, x_2, \dots, x_i, \dots, x_n)$, where x_i is the i th image. The n 2D images are learned from the basic CNN to

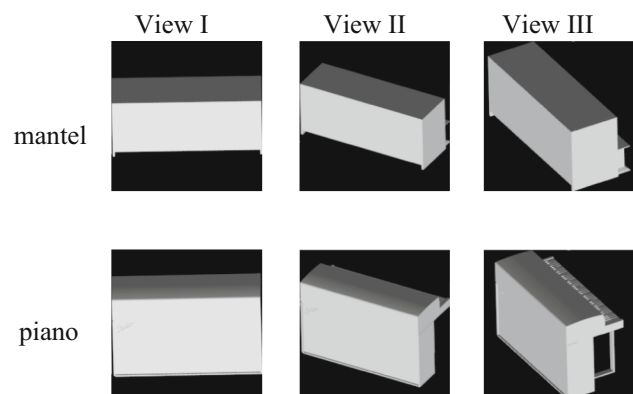


Fig. 2 Comparison of details from different categories

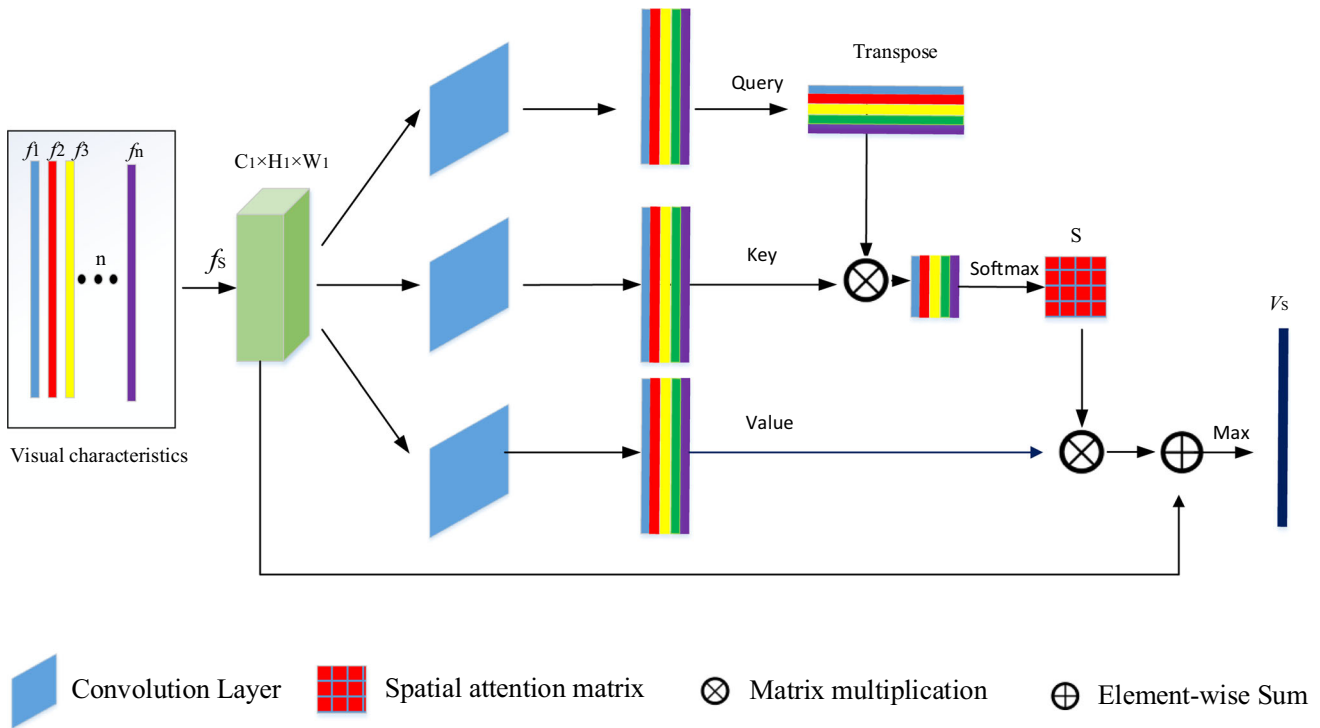


Fig. 3 View space attention block (VSAB). The key-pair query and the key assign different importance to each feature within the value of the spatial branch of the original feature, by exploring the spatial relationship between any two pixels of the view’s feature, to capture the discriminatory details within the view

a different visual feature $(f_1, f_2, \dots, f_i, \dots, f_n)$, where f_i is the visual feature of the i th image, $f_i \in R^{C \times H \times W}$. The overall visual features of n represent $f_S \in R^{C_1 \times H_1 \times W_1}$, as shown in Fig. 3. We first use two convolutional layers to generate two new feature maps, Query and Key, which can be expressed as

$$\text{Query} = Z(f_S), \tag{1}$$

$$\text{Key} = T(Z(f_S)), \tag{2}$$

where Z is the convolutional layer with a convolution kernel size of 1×1 , $(\text{Query}, \text{Key}) \in R^{C_1 \times H_1 \times W_1}$, and we reshape it as $R^{C_1 \times N}$, where $N = H_1 \times W_1$ is the space size of the overall feature f_S and T is the transpose operation. We multiply the Query and Key matrices, and apply a softmax layer to calculate the spatial attention matrix $S \in R^{N \times N}$, where

$$S_{ij} = \frac{\exp(\text{Query}_i \cdot \text{Key}_j)}{\sum_{i=1}^N \exp(\text{Query}_i \cdot \text{Key}_j)}. \tag{3}$$

S_{ij} is the spatial attention weight matrix obtained from VSAB, which measures the correlation between the i th and j th positions within a view. The larger the weight, the more similar it is. The softmax function ensures that the total weight of view features is 1.

In the second stage, we feed the overall characteristics f_S into the original feature space branch, whose features have the same resolution as the input and are used to preserve the original feature information. After convolution, a new feature mapping value $\in R^{C_1 \times H_1 \times W_1}$ is generated. Similarly, we set it to $C_1 \times N$ and multiply it by the spatial attention matrix S . This stage assigns an importance to each location in the original feature space branch value, focusing on the salient features in each view, which can be characterized as

$$P_S = f_S + \theta \sum_{i=1}^N (\text{Value}_i \cdot S). \tag{4}$$

We introduce a scale parameter θ , θ adaptive control to obtain the spatial attention feature, which is initialized to zero and gradually learns to assign more weight. The input features f_S are summed to ensure that the information richness learned by the features after VSAB is no less than that of the original input features. According to Eq. (4), the spatial feature of each region in views can adaptively learn the surrounding information context through VSAB, and more accurately distinguish the region through the spatial attention moment matrix S , thus avoiding some irrelevant information.

The third stage uses the max pooling clustered view feature,

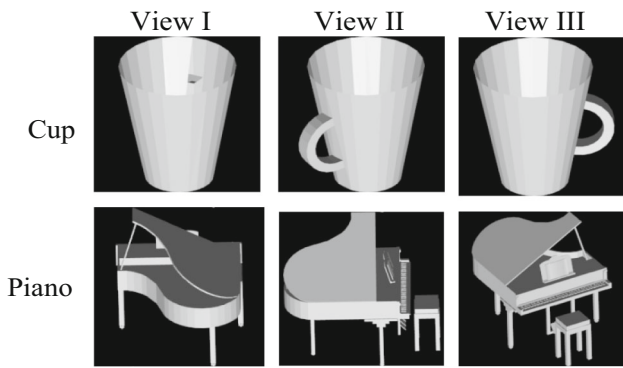


Fig. 4 Comparison of details from different perspectives

$$V_S = \text{Max}(P_S), \tag{5}$$

where Max is the max pooling operation. We gather VSAB features into a global descriptor to obtain the spatial shape descriptor V_S view channel attention block.

4.3 View channel attention block

After capturing the location relationship within a view, we want to obtain the relationship between views. These views have different characteristics with different contributions which will affect recognition accuracy. To learn the relative importance of views can better represent 3D shape descriptors and improve classification performance [34]. Figure 4 shows an example of three perspectives (View I, View II, View III) of the dataset category cup and category

piano after rendering the original 3D object. The purpose of using the view channel attention module is to find the relevance of multiple views, so as to focus on the important views. Two examples in View I ignore the two key features of cup handles and piano stools, but show them in the View II and View III. By emphasizing important perspectives and suppressing insignificant ones through VCAB, these more discriminatory views (View II, View III) respond more strongly (Fig. 5).

Therefore, we incorporate a channel attention mechanism in the pooling of visual features from different perspectives and propose that VCAB studies the correlation between different views. As shown in Fig. 5, the view channel attention block (VCAB) is divided into two phases. The first phase, extraction, obtains a channel attention weight by compressing spatial information from different perspectives, assigns each view an importance weight, and scales feature attention weights. The second phase, fusion, combines these view features in a global descriptor through a convolutional layer to obtain a view channel descriptor.

For the visual characteristics of n perspectives $(f_1, f_2, \dots, f_i, \dots, f_n)$, we first express their visual characteristics with the general feature $f_C \in R^{C_2 \times H_2 \times W_2}$. To find the dependencies between these perspectives, let us make $C_2 = n$. We need a channel descriptor to represent these overall characteristics, that is, to compress the spatial information of the overall characteristics of n perspectives into a single number. Thus, the visual feature f_c with a space size of $H_2 \times W_2$ shrinks to a channel vector $g \in R^n$,

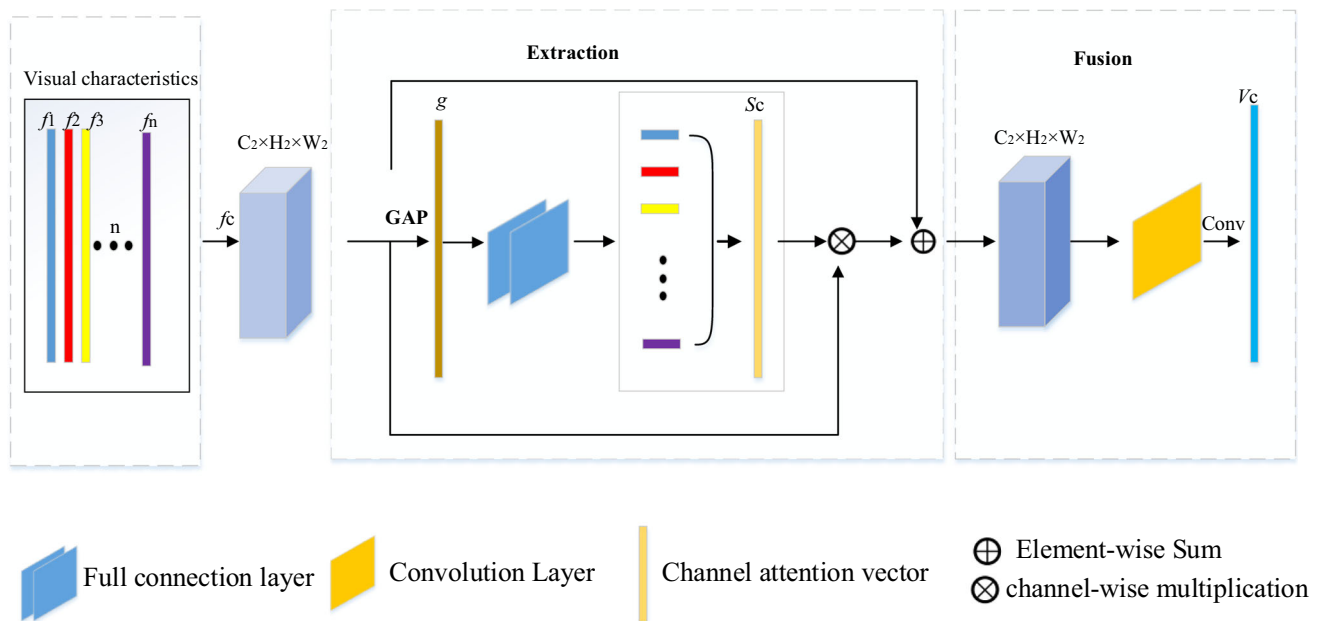


Fig. 5 View channel attention block. By compressing channel information from different perspectives, channel attention vectors are generated, the characteristics of each view are scaled with attention weights, and the correlation between the target perspectives is found, so that the distinguished perspectives in all perspectives are emphasized

$$g = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W f(i, j). \quad (6)$$

For the first step in the extraction phase, the overall feature f_c converts into an $n \times H_2 \times W_2$ size and compresses it to an $n \times 1 \times 1$ channel vector with global spatial information. The information for each channel is represented by its global average pooling (GAP).

After aggregating global spatial information for a channel, we must fully capture the relationship between view channels. A function to achieve this must meet two criteria. First, it must be able to measure the importance of each view and, specifically, to learn the interaction between views. Second, it must be able to estimate the content distinction of each perspective because we want to change the consistency of contributions from multiple perspectives, emphasize multiple beneficial view information, suppress irrelevant view information, and enable all views to contribute to different degrees to the characteristics of 3D objects based on their attention distribution weights. To meet these criteria, we use a view selection mechanism with two layers of full connection and one layer of ReLU, and use sigmoid function activation to calculate the attention vector S_c of the view channel,

$$S_c = \sigma(W_2 \delta(W_1 g)). \quad (7)$$

The two fully connected layers are a reduced-dimension layer and an increased-dimension layer. The attenuation ratio r of the reduced-dimension layer is assigned to n , which normalizes the number of view channels. W_1 and W_2 are parameters of dimension-reducing and dimension-increasing layers, $W_1 \in R^{n \times n}$ and $W_2 \in R^{n \times n}$, δ is a ReLU activation function, and σ is a sigmoid function that maps its output to $(0, 1)$ to obtain the channel attention weight vector S_c . The overall feature f_c of n perspectives is scaled, the size of which is $n \times 1 \times 1$. Attention weight vector S_c and the overall feature f_c of n perspectives are multiplied by an element-wise to obtain P , which can be expressed as

$$P = S_c \cdot f_c, \quad (8)$$

where ' \cdot ' represents channel-wise multiplication.

To ensure that the information richness of the learning view feature with the channel attention block is not less than the original input feature, the overall features f_c of the view are added to the final result,

$$P_C = P + f_c. \quad (9)$$

After feature extraction from the first phase of extraction, the second phase is the fusion of view features, which are combined into a global descriptor by the *Conv* operation to obtain the channel shape descriptor V_c :

$$V_c = \text{Conv}(P_c), \quad (10)$$

where *Conv* is a convolutional layer with core size $1 \times n$. Using a $1 \times n$ convolution kernel equivalent to an n -view window, the view channel shape descriptor V_c is obtained by sliding the view window and fusing the n -view angle features.

4.4 Prediction recognition results

In the first part of the MVDAN framework, we use ResNet [9] as our basic CNN model, and we also use mature networks such as VGG-M [7], DenseNet [10], and ResNeXt [35] to experiment. We remove the last full connection layer from the original ResNet [9], and join the proposed dual attention block of VSAB and VCAB. The modules work in parallel to obtain the corresponding view space and view channel descriptors, which are combined to form the final 3D shape descriptor, which eventually obtains the predictive recognition and classification results of 3D objects through a fully connected layer.

5 Experimental results and discussion

5.1 Dataset

The most widely recognized dataset for 3D shape classification is the Princeton ModelNet [36] series. We evaluated our approach on the ModelNet40 dataset, which includes 12,311 3D CAD models from 40 categories, with 9843 training models and 2468 test models. Because the numbers of samples vary by category, the experimental accuracy index is overall accuracy (OA) for each sample and (AA) for each category, OA is the percentage of correct predictions in all samples, and AA is the average accuracy for each category.

5.2 Implementation details

In all our experiments, there are two stages of training. The first stage only classifies a single view for fine-tuning the model, in which the dual attention block is removed. The second stage of training adds dual attention blocks to train all the views of each 3D model and performs joint classification for the views, and this is used to train the entire classification framework. When testing, only the second stage is used to make predictions. We use VGG-11 [7] pre-trained on the ImageNet [28] dataset as the first part of the basic CNN model in our network. We run our experiments on a computing node running Windows 10 system with an Intel core i7-8700K CPU at 3.70 GHz, 60-GB RAM, and an NVIDIA GTX 1080Ti graphics processing unit (GPU). We use the blender software for windows to generate

Table 1 Influence of the basic CNN model

Number	Network	OA	AA	Time (min)
12 view	VGG-11 [7]	95.17	92.72	103 + 199
12 views	DenseNet [10]	95.66	93.02	166 + 477
12 views	ResNet-50 [9]	96.63	95.51	155 + 283
12 views	ResNeXt-50 [35]	96.68	95.05	234 + 475
12 views	ResNet-50+ SE [31]	95.74	94.77	162 + 339
12 views	ResNet-50+ CBAM [32]	96.11	94.48	229 + 460
12 views	ResNet-50+ SC [38]	96.43	94.63	175 + 358

Bold values represent the best performance

Time is expressed as time spent in the first and second phases

multi-view rendering, and use pytorch1.2 and cuda10.0 for deep learning. We initialize the learning rate to 0.0001, using the Adam [37] optimizer for both phases. Learning rate decay and L2 regularized weight decay are used to reduce model overfitting. The weight decay parameter is set to 0.001. For single-GPU training, the epoch of the two stages will be 30. If dual-GPU training is used, the convergence speed of the model will be faster when the batch size is twice that of single-GPU training. Therefore, the number of training epochs in the first stage is adjusted to 10, and the number of training sets in the second stage is adjusted to 20. Single-GPU training time can be seen in Table 1, and time is expressed as the time spent in the first and second phases.

5.3 Influence of basic CNN model

View-based 3D object recognition methods [11, 25, 26] use VGG-M [7] as the basic CNN model, so we used VGG-M [7] as a comparison. Good neural networks can significantly improve performance in many applications. A variety of architectures have been proposed for large-scale CNNs, such as ResNet-50 [9], DenseNet [10], and ResNeXt [35]. We designed different running settings on ModelNet40 to connect the dual attention module to different basic CNN models to study the suitability of MVDAN to different network models. In particular, recent work has shown that [31, 32, 38] can easily be embedded in various networks to improve performance using attention mechanisms. We used single-GPU training with a learning rate of 0.0001.

As Table 1 shows, with the same number of perspectives, VGG-11 maintained 95.17% instance accuracy with the minimum training time (103 + 199 min). DenseNet [10] is deep, training time was longer, and accuracy improvement was not obvious. ResNet-50 [9] performed best in all CNN models, achieving 96.63% and 95.51%

accuracy on OA and AA, respectively, without much training time. The performance of ResNeXt-50-32x4d [34] was close to that of ResNet-50 [9], but training took the most time. Therefore, ResNet-50 [9] was chosen as the basic CNN model. We experimented with attention-based methods SENet [31], CBAM [32], and SCNet [38] embedded in the ResNet-50 [9] network framework, with less-than-ideal results. When attention is introduced in the underlying CNN model in conjunction with our proposed dual attention module, too much attention can degrade performance. ResNet [9] is seen as the best choice for the basic CNN model in the MVDAN framework.

5.4 Influence of number of views

To investigate the impact of the number of perspectives on classification performance, we changed the number of views for training and testing. We compared with MVCNN [11], MHBN [25], RCPCNN [26] and GVCNN [34], using 3, 6, and 12 views. The accuracy of the comparison method was obtained from Yu et al. [25], Wang et al. [26], and Yang et al. [39]. Table 2 shows OA from different perspectives on the ModelNet40 dataset.

Table 2 shows that MVDAN performed the best from all views. Overall accuracy reached 96.1%, 96.3%, and 96.6% in 3, 6, and 12 perspectives, respectively, for increases of 2.6%, 2.2%, and 2.3% over the relation network [39]. It is worth noting that when the number of views is further increased from 6 to 12, the performance of most methods [11, 25, 26] decreases. Our approach achieves the best performance at 12 views thanks to the relative importance of VCAB learning views, which enables the network to robustly represent the relationships between views as their number increases.

Table 2 Influence of the number of views

Methods	Number of views		
	3 views	6 views	12 views
MVCNN [11]	91.3	92.0	91.5
RCPCNN [26]	92.1	92.2	92.2
GVCNN [34]	–	–	92.6
MHBN [25]	93.8	94.1	93.4
Relation network [39]	93.5	94.1	94.3
MVDAN (Ours)	96.1	96.3	96.6

Bold values represent the best performance

Overall accuracy is expressed as a percentage

5.5 Comparison with other pooling methods

We investigate the effect of our dual attention module on classification performance. For fairness, we chose the same VGG-M [7] as Yu et al. [25] as a benchmark CNN model.

We removed the dual attention module and compared it to sum pooling, max pooling, bilinear pooling [40], improved bilinear pooling [41], log-covariance pooling [42], and harmonized bilinear pooling [25]. Table 3 shows the overall and category accuracy of 12-view different pooling methods on the ModelNet40 dataset. Table 3 shows that our dual attention module performs best, with 95.17% and 92.72% accuracy on OA and AA, respectively. Performance increased further with ResNet-50 [9] as the benchmark CNN model, as seen in Table 4.

5.6 3D object classification

We compare our methods to SPH [43] and LFD [44], which use manual descriptors, and then to voxel-based methods ShapeNets [16], VoxNet [15], and Pointgrid [18]; point-based methods PointNet [19], PointNet++ [20], Mo-Net [22], and 3D Capsule [45]; and view-based methods MVCNN [11], MVCNN-MultiRes [46], Relation Network [39], RCPCNN [26], GVCNN [34], and MHBN [25].

As Table 4 shows, view-based methods in all methods performed best. Using VGG-M [7] as the base model, our method had a higher classification accuracy than other view-based methods and performed best using ResNet-50 [9], with 96.6% and 95.5% correct classification accuracy on OA and AA, respectively. The excellent performance of our approach is due to the following reasons. MVDAN contains the dual attention modules VSAB and VCAB. VSAB explores the spatial relationships between any two pixels of a view’s characteristics to capture the discriminatory details within the view, and VCAB looks for correlation between target views, assigns weights to views, and facilitates a focus on discriminatory views. Compared to MVCNN [11], MVDAN combines these two modules, emphasizing the key features within and between views to

Table 4 Comparisons of performance with state-of-the-art methods

Methods	Input modality	ModelNet40	
		OA	AA
SPH [43]	Handcraft	–	68.2
LFD [44]	Handcraft	–	75.5
3D ShapeNets [16]	Volume	–	77.3
VoxNet [15]	Volume	–	83.0
Pointgrid [18]	Volume	92.0	88.9
PointNet [19]	Points	89.2	76.2
PointNet++ [20]	Points	91.9	–
3DCapsule [45]	Points	92.7	–
Mo-Net [22]	Points	92.4	90.3
MVCNN [11]	12 views	92.1	89.9
MVCNN-MultiRes [46]	Multi-resolution Views	93.8	91.4
Relation network [39]	6 views	94.1	–
	12 views	94.3	92.3
RCPCNN [26]	12 views	93.8	–
GVCNN [34]	8 views	93.1	–
	12 views	92.6	–
MHBN [25]	6 views	94.1	92.2
	12 views	93.4	–
MVDAN (VGG-M [1])	12 views	95.2	92.7
MVDAN (ResNet-50 [3])	12 Views	96.6	95.5

Bold values represent the best performance
 Numbers are reported in percentage terms. Our MVDAN achieves the best performance consistently

form a distinguishing global descriptor, which is better than treating all views equally.

6 Conclusions

In this paper, we proposed an MVDAN for 3D object recognition. Compared with the traditional methods, our proposed MVDAN framework no longer treats each view equally, but uses dual attention blocks to consider the

Table 3 Comparison with other pooling methods

Methods	Overall accuracy (OA)	Average accuracy (AA)
Sum pooling	88.04	85.20
Max pooling	87.35	85.65
Bilinear pooling [40]	87.03	84.81
Improved BP [41]	93.23	91.21
Log-covariance pooling [42]	93.03	90.57
Harmonized bilinear pooling [25]	94.12	92.23
MVDAN (VGG-M [7])	95.17	92.72

Bold values represent the best performance
 Accuracy is expressed as a percentage

correlation between the views within and between shapes. In order to better evaluate our framework, we conducted a series of experiments to study the influence of different components in our method. The experimental results and the comparison with the latest methods prove the excellence of the method, which achieved state-of-the-art performance on the ModelNet40 [29]. This method can be widely used in automatic driving, augmented reality, and reverse engineering, as well as automatic sorting, handling, assembly, and the detection of robots.

In future work, we will conduct more experiments with datasets collected from real environments, focusing on the robustness of our method and the suppression of noise components, and use possible image preprocessing methods such as wavelet transform to improve our classification results.

Authors' contributions All the authors made significant contributions to this work. WW and YC conceived and designed the experiments; YC performed the experiments; WW, YC and TW analyzed the data; and WW contributed analysis tools. All the authors have read and agreed to the published version of the manuscript.

Funding Financial support for this work was provided by the Natural Science Foundation of Shanghai under Grant No. 19ZR1435900.

Availability of data and materials The dataset used is provided by CVPR2015 at <http://modelnet.cs.princeton.edu/>.

Code availability The codes and models will be available at <https://github.com/caiyu97/MVCNN>.

Declarations

Conflict of interest The authors declare no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Grenzdörffer T, Günther M, Hertzberg J (2020) YCB-M: a multi-camera RGB-D dataset for object recognition and 6DoF pose estimation. In: IEEE international conference on robotics and automation (ICRA). IEEE, pp 3650–3656
- Yu D, Ji S, Liu J et al (2021) Automatic 3D building reconstruction from multi-view aerial images with deep learning. *ISPRS J Photogramm Remote Sens* 171:155–170
- Kästner L, Frasineanu V C, Lambrecht J (2020) A 3d-deep-learning-based augmented reality calibration method for robotic environments using depth sensor data. In: IEEE international conference on robotics and automation (ICRA). IEEE, pp 1135–1141
- Adikari SB, Ganegoda NC, Meegama RGN et al (2020) Applicability of a single depth sensor in real-time 3D clothes simulation: augmented reality virtual dressing room using kinect sensor. *Adv Hum Comput Interact*. <https://doi.org/10.1155/2020/1314598>
- Yang Y, Fang H, Fang Y et al (2020) Three-dimensional point cloud data subtle feature extraction algorithm for laser scanning measurement of large-scale irregular surface in reverse engineering. *Measurement* 151:107220
- Liu CH, Lee P, Chen YL et al (2020) Study of postural stability features by using kinect depth sensors to assess body joint coordination patterns. *Sensors* 20(5):1291
- Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint. <https://arxiv.org/abs/1409.1556>
- Szegedy C, Liu W, Jia Y et al (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
- 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
- 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
- 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
- Özbay E, Çınar A (2019) A comparative study of object classification methods using 3D zernike moment on 3D point clouds. *Traitement du Signal* 36(6):549–555
- Li Q, Cheng X (2018) Comparison of different feature sets for TIs point cloud classification. *Sensors* 18(12):4206
- Chen C, Li X, Belkacem AN et al (2019) The mixed kernel function SVM-based point cloud classification. *Int J Precis Eng Manuf* 20(5):737–747
- Maturana D, Scherer S (2015) Voxnet: a 3d convolutional neural network for real-time object recognition. In: IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE, pp 922–928
- 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
- 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
- Le T, Duan Y (2018) Pointgrid: a deep network for 3d shape understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 9204–9214
- Qi C R, Su H, Mo K et al (2017) PointNet: deep learning on point sets for 3D classification and segmentation. In: 30th IEEE conference on computer vision and pattern recognition, CVPR, pp 77–85
- Qi C R, Yi L, Su H et al (2017) PointNet++: deep hierarchical feature learning on point sets in a metric space. In: 31st Annual

- conference on neural information processing systems, NIPS 2017, pp 5100–5109
21. Achlioptas P, Diamanti O, Mitliagkas I et al (2018) Learning representations and generative models for 3d point clouds. In: International conference on machine learning, pp 40–49
 22. Joseph-Rivlin M, Zvirin A, Kimmel R (2019) Momen(e): flavor the moments in learning to classify shapes. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
 23. Zhao H, Jiang L, Fu CW et al (2019) Pointweb: enhancing local neighborhood features for point cloud processing. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5565–5573
 24. Lin H, Xiao Z, Tan Y et al (2019) Justlookup: one millisecond deep feature extraction for point clouds by lookup tables. In: 2019 IEEE international conference on multimedia and expo (ICME). IEEE, pp 326–331
 25. 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
 26. Wang C, Pelillo M, Siddiqi K (2019) Dominant set clustering and pooling for multi-view 3d object recognition. arXiv preprint. <https://arxiv.org/abs/1906.01592>
 27. Xie Y, Zhang Y, Gong M et al (2020) MGAT: multi-view graph attention networks. *Neural Netw* 132:180–189
 28. Deng J, Dong W, Socher R et al (2009) Imagenet: a large-scale hierarchical image database. In: IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
 29. Jerhotova E et al (2011) Biomedical image volumes denoising via the wavelet transform. In: Applied biomedical engineering, pp 435–458
 30. Jaderberg M, Simonyan K, Zisserman A (2015) Spatial transformer networks. In: Advances in neural information processing systems, pp 2017–2025
 31. Hu J, Shen L, Sun G (2018) Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7132–7141
 32. Woo S, Park J, Lee JY et al (2018) Cbam: convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV), pp 3–19
 33. Phong BT (1975) Illumination for computer generated pictures. *Commun ACM* 18(6):311–317
 34. 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
 35. Xie S, Girshick R, Dollár P et al (2017) Aggregated residual transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1492–1500
 36. The Princeton ModelNet. <http://modelnet.cs.princeton.edu/>. Accessed 18 Jan 2021
 37. Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: International conference on learning representations, pp 1–13
 38. Liu JJ, Hou Q, Cheng MM et al (2020) Improving convolutional networks with self-calibrated convolutions. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10096–10105
 39. Yang Z, Wang L (2019) Learning relationships for multi-view 3D object recognition. In: Proceedings of the IEEE international conference on computer vision, pp 7505–7514
 40. Lin T Y, RoyChowdhury A, Maji S (2015) Bilinear CNN models for fine-grained visual recognition. In: Proceedings of the IEEE international conference on computer vision, pp 1449–1457
 41. Lin T Y, Maji S (2017) Improved bilinear pooling with CNNs. arXiv preprint. <https://arxiv.org/abs/1707.06772>
 42. Ionescu C, Vantzos O, Sminchisescu C (2015) Matrix back-propagation for deep networks with structured layers. In: Proceedings of the IEEE international conference on computer vision, pp 2965–2973
 43. Kazhdan M, Funkhouser T, Rusinkiewicz S (2003) Rotation invariant spherical harmonic representation of 3 d shape descriptors. *Symposium on geometry processing* 6:156–164
 44. Chen DY, Tian XP, Shen YT et al (2003) On visual similarity based 3D model retrieval. In: *Computer graphics forum*, vol 22, no 3. Blackwell Publishing, Inc, Oxford, pp 223–232
 45. Cheraghian A, Petersson L (2019) 3dcapsule: extending the capsule architecture to classify 3d point clouds. In: IEEE winter conference on applications of computer vision (WACV). IEEE, pp 1194–1202
 46. Qi CR, Su H, Nießner M et al (2016) Volumetric and multi-view CNNs for object classification on 3d data. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5648–5656
- Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.