IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 28, 2019, accepted August 7, 2019, date of publication August 12, 2019, date of current version August 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2934650

Deeply Exploiting Long-Term View Dependency for 3D Shape Recognition

YONG XU[1,2,3], CHAODA ZHENG[1], RUOTAO XU[1], AND YUHUI QUAN[1,4]

[1]School of Computer Science and Engineering, South China University of Technology, Guangzhou 510000, China
[2]Peng Cheng Laboratory, Shenzhen 518000, China
[3]Communication and Computer Network Laboratory of Guangdong, Guangzhou 510000, China
[4]Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, Guangzhou 510000, China

Corresponding author: Yuhui Quan (csyhquan@scut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61672241, Grant 61602184, Grant 61872151, and Grant U1611461, in part by the Natural Science Foundation of Guangdong Province under Grant 2016A030308013 and Grant 2017A030313376, in part by the Science and Technology Program of Guangzhou under Grant 201707010147 and Grant 201802010055, in part by the Guangdong Provincial Engineering and Technology Research Center of Big Data Analysis and Processing under Grant 20140904-160, and in part by the Fundamental Research Funds for the Central Universities under Grant x2js-D2181690.

ABSTRACT Recognition of 3D shapes is a fundamental task in computer vision. In recent years, view-based deep learning has emerged as an effective approach for 3D shape recognition. Most existing view-based methods treat the views of an object as an unordered set, which ignores the dynamic relations among the views, *e.g.* sequential semantic dependencies. In this paper, modeling the views of an object by a sequence, we aim at exploiting the long-term dependencies among different views for shape recognition, which is done by constructing a sequence-aware view aggregation module based on the bi-directional Long Short-Term Memory network. It is shown that our view aggregation module not only captures the bi-directional dependencies in view sequences, but also enjoys the robustness to circular shifts of input sequences. Incorporating the aggregation module into a standard convolutional network architecture, we develop an effective method for 3D shape classification and retrieval. Our method was evaluated on the ModelNet40/10 and ShapeNetCore55 datasets. The results show the encouraging performance gain from exploiting long-term dependencies in view sequences, as well as the superior performance of our method compared to the existing ones.

INDEX TERMS 3D shape recognition, long-term dependency, multi-view deep learning, view aggregation.

I. INTRODUCTION

Understanding 3D objects has been a fundamental problem since the establishment of computer vision, with a broad spectrum of applications including multimedia [1], augmented reality [2], [3], entertainment [4], robotics [5], [6], autonomous driving [7]–[10], 3D reverse engineering [11], [12], medical imaging [13], [14], and monitoring [15]. Due to the limited availability of 3D data, the early works focus on either theories of 3D representation or methods of image-based object recognition. Until recent years, 3D data has become ubiquitous with the rapid development of 3D object acquisition hardware (*e.g.* 3D scanners) as well as 3D modeling software (*e.g.* computer graphics), opening up opportunities to practical 3D object recognition. Compared to the 2D cases in images, 3D objects contain additional cues of 3D shapes, which are very essential and crucial for visual understanding. Therefore, a majority of existing works on 3D object recognition are devoted to building 3D shape features for recognition (*i.e.* 3D shape recognition).

Inspired by the great success of deep learning in image classification [16], [17], many approaches (*e.g.* [18]–[22]) to 3D shape recognition have been proposed based on neural networks (NNs). These approaches consume different formats of 3D data in NNs, such as point clouds, voxelized data, and multiple views of objects. Among these approaches, the ones that accept multiple views of objects as NN's input, which are called view-based approaches, have shown much more encouraging results than other types of approaches; see *e.g.* [21]–[25]. Besides, view-based approaches have

The associate editor coordinating the review of this article and approving it for publication was Aysegul Ucar.

many advantages, *e.g.* leveraging power of modern image-based convolutional networks (CNNs), sharing similarity with human perception, circumventing any geometric representation artifacts [21], etc.

Generally, view-based approaches pass multiple 2D views of 3D objects to image-based CNNs and aggregate the results for recognition. The key part in this pipeline is how to utilize the rich relations among different views to aggregate the features from different views for effective recognition, which is also the element that distinguishes view-based methods from traditional CNN-based image classification. Most existing view-based approaches (*e.g.* [21], [23]–[25]) aggregate view features by treating them as an unordered set of features, without considering the dynamics existing in the views (*e.g.* smooth changes of sequential views). As a result, the generated shape descriptors mainly encode the 'static' relations of views, and the relations on dynamics of views (*e.g.* sequential changes of views) have not been well exploited.

In many scenarios, the views are generated by a sequential process, *e.g.* moving a camera around the object. Therefore, the views may contain many sequential or temporal semantic dependencies. In this paper, we model the views as a sequence instead of an unordered set, which provides opportunities to utilize such rich dependencies among views. An interesting observation on the view sequence of an object is that, long-term dependencies may exist in the sequence. For instance, when looking around a car, the view taken from the left side is not only related to its neighboring views, but also closely related to that taken from the right side due to the symmetry of the car. See Fig. 1 for an illustration of this example. In addition to symmetries of objects, similar parts or similar patterns of objects, as well as smooth changes of both viewpoints and object surfaces, are the main sources of the long-term dependencies in view sequences.

The long-term dependencies in view sequences are very useful to visual recognition. For instance, people can optimize the trajectory of viewpoints to use views as few as possible for understanding objects; and the next unobserved view may be easily recognized or even created based on the sequence of observed views. To exploit the long-term dependencies of view sequences for 3D shape recognition, we propose a sequence-aware view aggregation module based on the long short-term memory (LSTM) [26]. An important property of the long-term dependencies of view sequences is the bi-directionality, which comes from the fact that a view sequence can be provided bi-directionally, *e.g.*, the camera can move clockwise or anticlockwise. In order to utilize this property for improving the effectiveness of aggregation, we use the bi-directional LSTM to construct our view aggregation module. With each view as the input of a bi-directional LSTM unit, we can effectively analyze the long-term dependencies of the view sequence along two directions. We also show that using bi-directional LSTM can bring robustness to circular shifts of view sequences which are often seen in real applications.

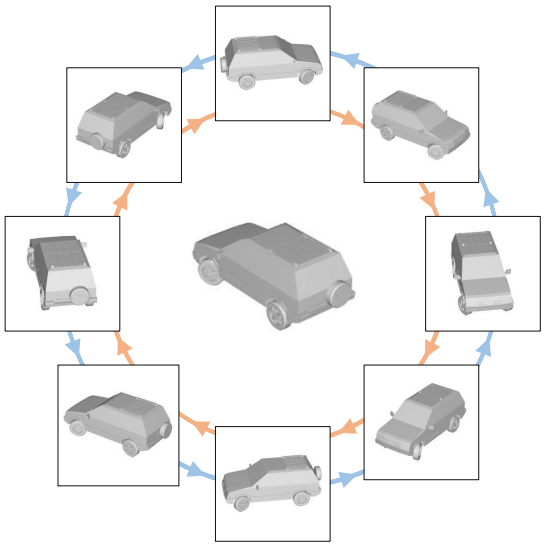


FIGURE 1. View sequence of a 3D car. It can be see that the view taken from the left side is not only related to its neighboring views, but also closely related to that taken from the right side due to the symmetry of the car. Note that the camera movement in generating the sequence can be clockwise or anticlockwise.

Building the proposed view aggregation module into a standard view-based CNN, we develop an effective method for 3D shape recognition. Leveraging the power of bi-directional LSTM, the proposed method is able to generate distinct yet robust shape descriptors. The proposed method was evaluated on the ModelNet40 and ModelNet10 datasets, with applications to classification and retrieval. The results show the encouraging performance gain from exploiting long-term dependencies in view sequences, as well as the superior performance of our method to the state-of-the-art ones. In particular, even using fewer views, our method can still outperform many existing methods.

In summary, the main contributions of this paper are two-fold. In methodology, we propose to treat the views of an object as a sequence and investigate the exploitation of the bi-directional long-term dependencies of view sequences for 3D shape recognition. The sequence-based treatment to views is nontrivial, as it enables the utilization of the dynamics among views (*i.e.* view images are taken in a dynamic process), which also distinguishes our method from the existing ones [21]–[25], [27], [28] that only focus on the static properties of views (*i.e.* similarities among views) by treating a view sequence as an unordered set. In practice, we propose an effective view-based CNN with a bi-directional LSTM-based aggregation module for 3D shape classification and retrieval. The proposed network has the advantages of analyzing the long-term semantic dependencies of view sequences along two directions, recognizing complex shapes, and achieving robustness to circular shifts of view sequence.

The rest this paper is structured as follows. Section II is for the literature review. Section III is devoted to the detailed description of the proposed method. Section IV presents the experimental evaluation. Section V concludes the paper.

II. RELATED WORK

The early approaches to 3D shape recognition focus on the design of handcrafted shape features. In recent years, inspired by the success of machine learning (particularly deep learning) in computer vision, many learning-based approaches have been proposed to learn adaptive shape descriptors from a set of 3D data. Our work follows this line of research. In this section, a brief review is given on the handcrafted features, followed by a detailed review on the learning-based methods.

A. HANDCRAFTED FEATURES

The classic shape descriptors, such as the statistical moments [29], Fourier descriptor [29], [30] and eigenvalue descriptor [31], are devoted to the global description of shapes. These methods suffer from the sensitivity to non-rigid transformations or topological changes. To overcome this weakness, some local geometric descriptors were proposed as the building blocks to form global shape features, *e.g.* spin images [32], shape context [33] and mesh HOG [34]. Nevertheless, such descriptors are not robust to local geometric deformations or perturbations. Recently, the diffusion-based approaches have emerged as a promising direction for shape description, which enjoy strong robustness to isometric deformations and small perturbations on surfaces. These methods model the geometric structure of shapes with a certain diffusion process, and the shape descriptors are built upon the associated diffusion operators, *e.g.* Discrete Laplace-Beltrami operator [35] and heat kernel signature [36], [37].

B. LEARNING-BASED METHODS

Embracing recent advances of deep learning and neural networks (NNs) in image classification (*i.e.* an analogous task to 3D shape recognition) [16], [17], most learning-based methods for 3D shape recognition are built upon NN architectures. There are mainly three formats of 3D data used in the NN-based methods, including points, voxels and views. According to the data format of NN's input, the NN-based methods can be classified into three categories: point-cloud based methods, voxel-based methods, and view-based methods. We focus more on the view-based methods in the literature review, as our method belongs to this type. It is noted that there is a group of learning-based approaches that take mesh surface as input by generalizing CNNs to non-Euclidean geometries (*e.g.*spectral CNNs [38], anisotropic CNNs [39]) or by using the handcrafted features of objects as input (*e.g.* [40]). These approaches are devoted to matching tasks, without published results on standard shape recognition. Thus, we omit this group of approaches in our literature review. It is also noted that a few approaches use two or more sources of 3D data for further improvement, *e.g.* both voxels and views are used in [41]. Last but not least, there are also some approaches built upon other machine learning techniques, *e.g.* multi-hypergraph learning [42].

1) POINT-CLOUD BASED METHODS

In contrast to image data which is row-column indexed, a point cloud (except for those computed from depth images) is generally a set of point coordinates with irregular organization and unordered structure, which hinders the trivial use of traditional image CNNs in point-cloud based methods. To address such a fundamental challenge arising from raw data, new NN architectures are needed. A pioneering work is PointNet [43], a permutation-invariant deep architecture that learns a spatial-encoding representation for each point and combines them into a global descriptor. In [44], the PointNet architecture is extended to a hierarchical version called PointNet++, which aims at better exploiting local structures of shapes by applying PointNet recursively on a nested grouping of the input point cloud. Another work with the same purpose of exploiting local shape structures is done in [45] via kernel correlation and graph pooling. The grouping scheme in PointNet++ is to implicitly exploit the spatial distribution of points. For explicit exploitation, the kd-Net [46] builds a kd-tree on the input point set and runs hierarchical feature extractions from the leaves to root. Due to the non-overlap partition by kd-tree, the kd-Net lacks of the overlapped receptive fields which are useful for recognition. To address this issue, Li *et al.* [47] proposed to replace the kd-tree with a self-organizing map (SOM) and perform k-NN search from points to SOM nodes, by which the receptive filed overlap can be controlled. Instead of directly dealing with point sets in the network, Simonovsky and Komodakis [20] proposed to structure the point cloud with a graph and apply a graph CNN to process the graph-structured data.

2) VOXEL-BASED METHODS

Voxels of 3D objects are a straightforward extension of pixels of 2D images, by which an object shape is represented as a volumetric binary occupancy grid. Unlike point cloud data, voxels are well indexed. Thus, the image-based CNNs can be easily extended to handle voxelized data. A seminal work can be traced back to 3D-ShapeNet [18], a volumetric convolutional deep belief network which expresses 3D shapes as a probability distribution of binary variables on a voxel grid. Another early attempt is Voxnet [19], which uses a shallow volumetric CNN with the volumetric probabilistic occupancy grid representation. The Voxnet architecture is jointed with an orientation estimation task in [48] for performance improvement. Since voxels are volumetric representations that can easily become computationally intractable, the above voxel-based NNs have to be shallow. To make use of the power of deep learning, Brock *et al.* [49] proposed a deep voxel-based CNN architecture which can be effectively and efficiently trained. With the same purpose, Riegler *et al.* [50] proposed to exploit the sparsity of voxelized data to enable deeper networks without reducing resolution. To anylase the shape distribution of 3D objects, [51] uses a VAE (variational auto-encoder) to reconstruct full 3D shapes from voxelized single-views. With the latent variables learned by the VAE,

FIGURE 2. Basic workflow of view-based deep learning methods.

numerical analysis of 3D ojbects can be well performed. While all aforementioned voxel-based methods are developed in the supervised setting, Wu *et al.* [52] proposed a unsupervised method to learn 3D shape descriptors using generative adversarial networks.

3) VIEW-BASED METHODS

Though voxel models can simplify the design of CNNs in comparison with point clouds, the computational burden of volumetric CNNs is still considerable. In this sense, the multi-view representation is a better choice for 3D data, as on each view, any image-based CNN can be directly applied. The advantages of multi-view representations are plenty [21]. First, by converting irregular 3D shapes to regular images, we can leverage pre-trained image-based CNNs that are powerful for recognition. Second, view-based methods can be applied to real objects with backgrounds. Third, multi-view representations share similarity with human perception, *e.g.* people look around an object to understand it by combining surfaces' information from multiple views. Forth, using multi-view representations can circumvent any geometric representation artifacts (*e.g.* non-manifold geometry, polygon soups, no interior). Last, multi-view representations can be generated from any other formats of 3D objects, such as point cloud, mesh, voxel, etc.

Most view-based methods share the basic flow path as follows. At first, multiple views of a 3D object are generated by projection rendering. Then, better representations (*i.e.* features) of different views are obtained by inputting the views to some modern image-based CNNs. Next, an aggregation (or called view pooling) module is run to aggregate the view-level CNN features into shape descriptors. Finally, the aggregated shape descriptors are fed to a classifier for recognition. See Fig. 2 for an illustration. In this pipeline, the most important part is the aggregation module, which distinguishes view-based methods from traditional CNN-based image classification.

The MVCNN [21] is one of the very first view-based methods, which uses CNNs with shared weights across different views for view feature extraction, as well as uses max pooling across views for view feature aggregation. The max pooling in MVCNN may cause significant loss of discriminative information in the aggregation of view features. To remedy this problem, Yu *et al.* [24] proposed a bilinear pooling which aggregates view features based on local patch similarity across views. To further exploit view similarity for better aggregation, Wang *et al.* [25] proposed to pool view features based on grouping, which is done by a repeated clustering and within-cluster pooling process. The idea of grouping features is also used in [23], in which the hierarchical view-group-shape feature aggregation pipeline is used. Due to the need for fixed-length input data, the aforementioned methods cannot deal with trajectories of arbitrary paths of camera movements. To break through this restriction, Johns *et al.* [53] proposed to build a pairwise view learning mechanism into NN, with the extension to predict the next best view. To guarantee the recognition performance when plenty of views of objects are missing, Kanezaki *et al.* [22] proposed to treat the view-point labels as latent variables, which are learned on unaligned objects with a joint pose estimation and shape recognition framework. View-based 3D shape recognition systems often suffer from high computation cost of projection rendering and feature matching. See [27] for the related acceleration techniques for view-based methods. It is worth mentioning that, instead of using the views taken from different viewpoints, Sfikas *et al.* [28], [54] used the panoramic view as the 2D representation of 3D objects. An approach [55] developed in parallel to ours shares similarities with the proposed method by using recurrent network to process view data. Different from ours, this approach use a one-way LSTM to aggregate shape features and an additional one-way LSTM to decode the predefined sequentialized class labels, with an attention mechanism applied to the network. In comparison, our method is much simpler yet effective, which only uses a bi-directional LSTM on view data without sequentializing class labels and using attention mechanism, and achieved better results on the test datasets. When preparing our manuscript, we noticed that two methods [56], [57] were developed in parallel to ours, which also process view sequences with LSTM. Compared to these two methods, ours has different motivations of using LSTM (*e.g.* symmetry of 3D objects). Furthermore, our method has simpler structures (*e.g.* no Siamese structure as [57] and no highway layer as [56]) but higher performance (see the experiments for the comparison).

III. PROPOSED METHOD

The architecture of the proposed NN for 3D shape recognition is illustrated in Fig. 3. The pipeline is as follows. First, a multi-view representation of a 3D object is generated as the input of the network, with each view as an image. Then, each view image is fed into a CNN for extracting the view feature. Next, we collect all view features as a sequence and pass them through a bi-directional LSTM network for view aggregation, with each view feature as the input of the LSTM unit at the corresponding time step. The LSTM unit at each time step outputs a new view feature which encodes the long-term dependency in the view sequence, followed by a fully-connected layer for predicting the corresponding class label. Finally, the classification results are combined to generate the final prediction. The details of each step are given in the following subsections. In particular, the sequence-based view aggregation module is our main focus.

For the convenience of presentation, we define the notations used in the following as follows. Light upper letters are used for constants (*e.g.* M), light lower letters for scalars (*e.g.* x), bold lower letters for column vectors

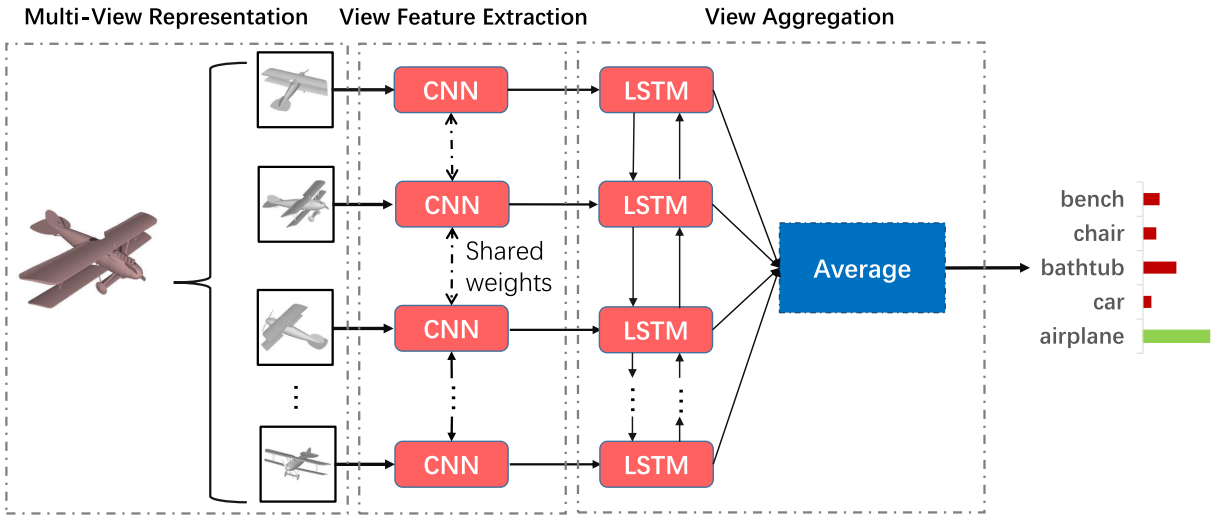


FIGURE 3. Architecture of the proposed view-based network for 3D shape recognition. Multiple views are first created from the input 3D object. Then a shared CNN is used to extract features from each view image. Next, a LSTM-based view aggregation module is run on the view features to generate higher-level representations of the views. Then new representations are then classified with fully-connected layers. Lastly, the classification results are combined by the average-pooling for the final prediction.

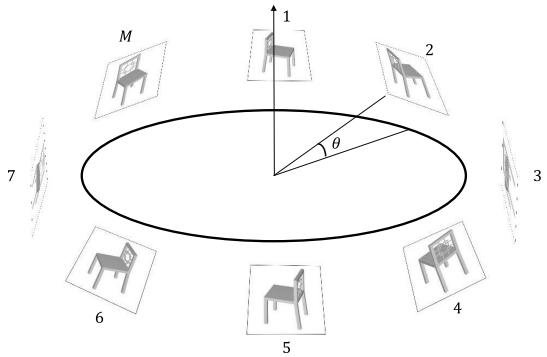


FIGURE 4. Viewpoint setting of the proposed method, with M virtual cameras evenly placed around the rotation axis and elevated by angle θ from the ground plane. The cameras are pointing to the centroid of the 3D shape when capturing image. In this example, we set $M = 8$ and $\theta = 30°$.

(*e.g.* $x = [x_1, \cdots, x_M]$), bold upper letters for matrices (*e.g.* $X = [x_1, \cdots, x_M]$), hollow letters for sets (*e.g.* $\mathbb{X} = \{X_1, \cdots, X_M\}$), and calligraphy letters for sequence (*e.g.* $\mathcal{X} = (X_1, \cdots, X_M)$).

A. MULTI-VIEW REPRESENTATION

In view-based methods, the first step is to obtain the multi-view representation of the given 3D object (which is usually stored as a polygon mesh). In our method, the multi-view representation is created by capturing images of size $N \times N$ (most existing methods set $N = 224$) from different virtual cameras placed at different viewpoints using projection rendering. The selection of viewpoints determines how well a multi-view representation can depict the original 3D shape. Following the standard setting in existing view-based methods (*e.g.* [21], [22], [25]), we use the camera setup illustrated in Fig. 4 to generate multi-view representations, where M cameras are evenly placed around the rotation axis and elevated by angle θ from the ground plane. In our implementation, we set $M = 12$, $\theta = 30°$ by default. The generated view images are picked up clockwise from a fixed position to form a view sequence, which is denoted by $(V_1, \cdots, V_M)$, where $V_m \in \mathbb{R}^{N \times N}$ is a view image, for $m = 1, \cdots, M$. This process is denoted by

$$\mathrm{MV}(\cdot; M, N, \theta) : \mathbb{X} \rightarrow (V_1, \cdots, V_M), \tag{1}$$

where $\mathbb{X}$ denotes a given 3D object.

B. CONVOLUTIONAL VIEW FEATURE EXTRACTION

The view images generated from multi-view representation are the images taken from the objects, which may be not robust and distinct for recognition. Thus, an improved representation is needed on each view image. Inspired by the recent success of VGG networks [16] on many image recognition tasks [17], we employ the VGG-11 network [16] with batch normalization as our base CNN, which is simple but sufficiently effective for extracting features from each view image. The VGG-11 network is mainly composed of eight 3×3 convolutional layers in the front and three fully-connected layers in the back. In our implementation, given a view image $V \in \mathbb{R}^{N \times N}$ as input, the output from the second last fully-connected layer is used as the view feature, which is a 4096-dimensional feature vector denoted by $d \in \mathbb{R}^{4096}$. Then, by sequentially feeding the view sequence $(V_1, \cdots, V_M)$ into the CNN, we obtain a sequence of convolutional view features which are denoted by $(d_1, \cdots, d_M)$. We denote such a process by

$$\mathrm{VGG}(\cdot, \omega) : (V_1, \cdots, V_M) \rightarrow (d_1, \cdots, d_M), \tag{2}$$

where $\mathrm{VGG}(\cdot, \omega)$ denotes the VGG-11 network with all its weights encoded in the parameter vector ω.

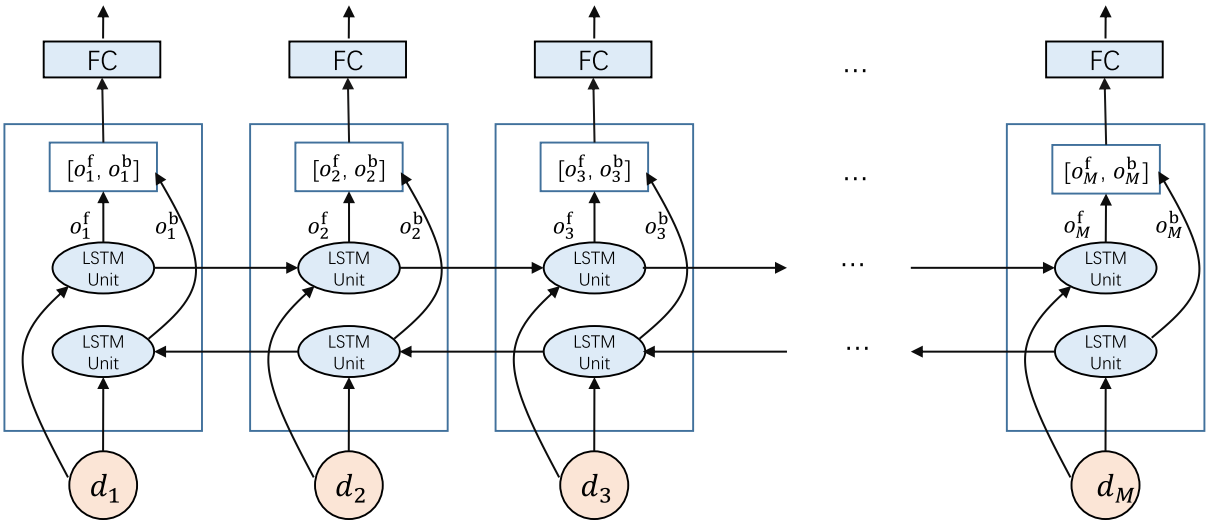


FIGURE 5. Illustration of the Bi-directional LSTM network used in our view aggregation module.

C. SEQUENCE-BASED VIEW AGGREGATION

The view features generated in the previous step need to be aggregated for classification. As discussed in Section I, each view image in the view sequence may have strong correlation with each other, which implies long-term dependencies in the view sequence as well as in the sequence of convolutional view features. Furthermore, the view sequence is undirected, implying bi-directionality on the long-term dependencies. To exploit such bi-directional long-term dependencies for aggregating view features, we employ the bi-directional LSTM [26], [58] as a sub-network. The bi-directional LSTM accepts the view feature sequence as input and output new features for each view which encodes the long-term dependency in the view sequence. It has been shown in many studies [59], [60] that the bi-directional LSTM network has the capability of learning the long-term dependency in sequential data.

The framework of the bi-directional LSTM sub-network in our method is illustrated in Fig. 5. In details, our bi-directional LSTM network is a single-layer recurrent network, accepting the view feature d_m as input at the mth time step. The LSTM layer is a big unit consisting of two independent LSTM subunits, one for forward sequential connection and the other for backward sequential connection. Given the sequence of view features $(d_1, \cdots, d_M)$ as input, the forward LSTM units are defined by the following equations [26]:

$$\begin{cases} i_m = \sigma(W_{\text{ii}} d_m + b_{\text{ii}} + W_{\text{hi}} h_{m-1} + b_{\text{hi}}) \\ f_m = \sigma(W_{\text{if}} d_m + b_{\text{if}} + W_{\text{hf}} h_{m-1} + b_{\text{hf}}) \\ g_m = \tanh(W_{\text{ig}} d_m + b_{\text{ig}} + W_{\text{hc}} h_{m-1} + b_{\text{hg}}) \\ o_m = \sigma(W_{\text{io}} d_m + b_{\text{io}} + W_{\text{ho}} h_{m-1} + b_{\text{ho}}) \\ c_m = f_m \odot c_{m-1} + i_m \odot g_m \\ h_m = o_m \odot \tanh(c_m) \end{cases} \tag{3}$$

for $m = 1, \cdots, M$, where $h_m, c_m, i_m, f_m, g_m, o_m \in \mathbb{R}^H$ denote the hidden state, cell state, input gate, forget gate, cell gate and output gate at the mth time step respectively, $W_{\text{ii}}, W_{\text{if}}, W_{\text{ig}}, W_{\text{io}} \in \mathbb{R}^{H \times 4096}$, $W_{\text{hf}}, W_{\text{hc}}, W_{\text{ho}} \in \mathbb{R}^{H \times H}$, $b_{\text{ii}}, b_{\text{if}}, b_{\text{ig}}, b_{\text{io}}, b_{\text{hi}}, b_{\text{hf}}, b_{\text{hg}}, b_{\text{ho}} \in \mathbb{R}^H$ are the weights of network to be learned, $\odot$ denotes the element-wise product, σ denotes the sigmoid function, and c_0 and h_0 are set to be zeros. The backward LSTM units can be defined in the same way, which are implemented by reversing the sequence as input. The output hidden states of the forward and backward LSTM units at the same time step are concatenated together as a new feature for the corresponding view, which is denoted by $f_m, m = 1, \cdots, M$. In our implementation, we set $H = 2048$ and thus the bi-directional output state generate a 4096-dimensional descriptor $f_m \in \mathbb{R}^{4096}$ for all m.

Due to the use of bi-directional LSTM, the new features of views are the aggregated versions of the original ones, which encode the long-term dependency in the view sequence and thus are very effective for recognition. Such features from each LSTM unit are then fed into a fully-connected layer with the softmax activation for class label prediction:

$$l_m = \text{softmax}(W_m f_m + b_m), \tag{4}$$

where l_m denotes the prediction vector of the mth view, $m = 1, \cdots, M$, and W_m, b_m are the weights to be learned. In short, the above aggregation process is formulated as

$$\text{LSTM}(\cdot, \psi) : (d_1, \cdots, d_M) \rightarrow \{l_1, \cdots, l_M\} \tag{5}$$

where $\text{LSTM}(\cdot, \psi)$ denotes the whole LSTM-based aggregation module, with all related network parameters encoded in the parameter vector ψ.

One possible issue of using recurrent networks such as LSTMs is the lack of robustness to the circular shift of input view sequence. In fact, the circular shift is often seen in the multi-view representation of view-based methods. For instance, in our setting where cameras are placed in a circle around an rotation axis, rotating the object around the rotation axis, or starting with a different camera when collecting

views, may generate a view sequence that is the circular shift of the original one. Fortunately, the bi-directional structure of our LSTM network can noticeably improve the robustness of our method to circular shifts of view sequences. Concretely, let $(d_1, d_2, \ldots, d_m, d_{m+1}, \ldots, d_M)$ denote the input sequence, and $o_m^{\mathrm{f}}, o_m^{\mathrm{b}}$ denote the forward output state and backward output state at the mth time step respectively. By definition, the forward state o_m^{f} carries the information from $(d_1, d_2, \ldots, d_m)$ while the backward one carries that from $(d_m, d_{m+1}, \ldots, d_M)$. In other words, the concatenated output state at each bi-directional time step covers the information of the whole sequence $\mathcal{X}$. It can be easily verified that, with any circle shift, though the information covered by o_m^{f} and o_m^{b} are changed respectively (implying sensitivity of one-way LSTM), the mth output state still covers the information of the whole sequence, which implies the robustness. The performance improvement benefited from the bi-directionality is also supported by our experimental results; see *e.g.* Table 5.

During test, the predictions from all the LSTM units are first aggregated by

$$l = \frac{1}{M}\sum_{m=1}^{M} l_m. \tag{6}$$

Then, the final class label is calculated by taking the index of the maximal element in l.

D. TRAINING

Based on the above descriptions, the whole process of our NN can be written as follows:

$$\mathcal{F}(\cdot; \psi, \omega, M) := \mathrm{LSTM}(\mathrm{VGG}(\mathrm{MV}(\cdot; M); \omega); \psi), \tag{7}$$

where $\mathcal{F}$ accepts a 3D object and outputs M view labels $\{l_1, \cdots, l_M\}$. In our implementation, the parameter M in multi-view representation is set to a predefined number. Given a set of training data $\mathbb{D} = \{(\mathbb{X}_k, z_k)\}_{k=1}^{K}$, where $\mathbb{X}_k$ is a 3D object with one-hot encoding class label $z_k \in \mathbb{R}^Q$ for Q classes, for $k = 1, \cdots, K$. We first generate the view sequence $\mathcal{V}_k$ on each 3D object $\mathbb{X}_k$ by

$$\mathcal{V}_k = \mathrm{MV}(\mathbb{X}_k; M), \tag{8}$$

for all k. During training, we do not average the M predicted labels. Instead, each of the M predicted labels is paired with the true label to compute the loss. Then the loss function of our NN is defined by

$$\mathcal{L}(\mathbb{D}; \psi, \omega) = \sum_{k=1}^{K} f(z_k, \mathcal{F}(\cdot; \psi, \omega, M)) + g_1(\omega) + g_2(\psi)$$

where $g_1(\cdot), g_2(\cdot)$ are the weight decay regularizations, and f is the cross-entropy loss defined by

$$f(x, \{y_1, \cdots, y_M\}) = -\sum_{m=1}^{M}\sum_{q=1}^{Q} x(q)\log(y_m(q)).$$

The parameters of our aggregation module encoded in ψ are initialized using Xavier Initializer [61] and learned from the training data set. Regarding the parameters of VGG-11 encoded in ω, due to the insufficiency of 3D data as well as the considerations on the computational burden in training, we initialize the parameters with the ones pretrained on the ImageNet ILSVRC 2012 dataset [17] and fine-tune them together with ψ.

E. COMPARISON WITH SIMILAR WORKS

We noticed that two approaches [56], [57] developed parallel to ours use a similar architecture with us. They both use a share CNN to extract view features and adopt a bidirectional LSTM network for view aggregation. However, the motivations of using LSTM especially bi-LSTM are not very clear in both works. Comparison of network architectures is demonstrated in table 1. Besides difference in choices of CNNs, [56] places a highway layer between the CNN and LSTM network to alleviate gradient vanishing problem as the network is relatively deeper. And [57], aiming at just object retrieval, constructs the CNN-BiLSTM into a siamese structure with the contrastive loss to further enhance the discriminative power of shape features. [57] and ours both simply apply average pooling to aggregate LSTM features while [56] uses a weighted average scheme where features at later time are assigned a higher weight.

It's worth mentioning the difference in LSTM output dimension. References [56] and [57] make it 2 * 128 and 2 * 512 respectively, and they are different with the corresponding CNN feature dimension(512 and 4096). In contrast, our LSTM output dimension is deliberately designed to be 2 * 2048, which is just equal to our CNN feature length. As view pooling cases have the nature of keeping the feature dimension unchanged, we do this because we don't want the view aggregation module to alter the feature dimension. Performance comparison of these three methods is demonstrated in section IV.

IV. EXPERIMENTS

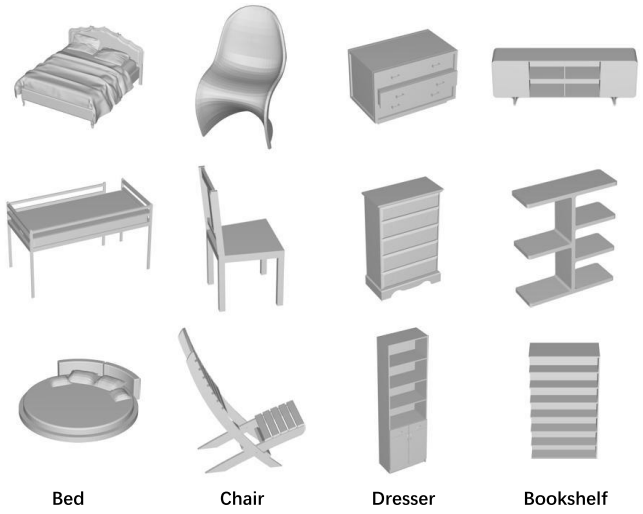
We evaluate the performance our method by applying it to 3D shape classification and retrieval. Three public widely-used datasets for the performance test of 3D shape recognition are employed in our evaluation, including

- ModelNet40 [18] which contains 12311 CAD models from 40 categories;
- ModelNet10 [18] which contains 4899 CAD models belonging to 10 categories;
- ShapeNetCore55 [62] which contains 51162 3D models categorized into 55 categories and 203 subcategories.

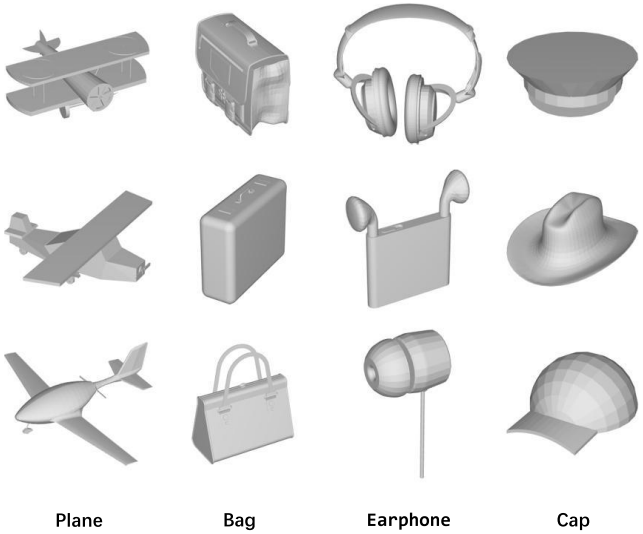
All the models from the above datasets are consistently aligned and satisfy the upright assumption. In Fig. 6 we show samples from the datasets. As can be seen, there are huge diversities among inner-class objects (*e.g.* the bookshelves and beds in Fig. 6a as well as the earphones in Fig. 6b) and high similarities of inter-class objects (*e.g.* the dresser and bookshelf in the bottom row of Fig. 6a).

TABLE 1. Architecture comparison.

	Ours	CNN + LSTM [56]	Siamese CNN-BiLSTM [57]
CNN Architecture	VGG-11 with BN	ResNet18	AlexNet
CNN feature Dim.	4096	512	4096
LSTM output Dim.	2 * 2048	2 * 128	2 * 512
Voting Scheme	average	weighted average	average
Dropout in LSTM	-	rate 0.5	-
Additional Comment	-	Highway network layer between CNN and LSTM	Siamese architecture



(a) ModelNet10/ModelNet40



(b) ShapeNetCore55

FIGURE 6. Some of the samples from ModelNet40/ModelNet10 and ShapeNetCore55 datasets. Models at each column are from the same category.

The implementation details of our method is as follows. The size of each view image is set to 224 × 224. The view-level features consumed by our view aggregation module are obtained from FC_{10} of VGG11 and thus the dimension of each feature vector is 4096. For the LSTM unit configuration, we set the size of the hidden state to be $H = 2048$ both for the forward unit and the backward unit. Then the descriptors generated from the output state of bi-directional LSTM unit is 4096-dimensional. The whole network was trained with a multi-stage manner. At the 1st stage, the VGG-11 module pre-trained on ImageNet ILSVRC 2012 dataset [17] is fine-tuned on multi-view images rendered from the corresponding 3D dataset. At 2nd stage, we initialize the CNN in our network using the fine-tuned VGG-11 in the 1st stage, and then train the network with the CNN fixed, only updating the bi-LSTM. At the 3rd stage, we update the CNN while fixing the parameters of bi-LSTM. At the last stage, the CNN and the LSTM are trained jointly to further improve the network performance. The number of epoch is set to 60 at each training stage. The learning rate is initialized by 0.01, 0.01, 0.001, 0.001 respectively at each stage. The stochastic gradient descend with momentum set to 0.9 is both adopted in above schemes.

A. 3D SHAPE CLASSIFICATION

In the classification experiments, the ModelNet10 and ModelNet40 datasets are used. For fair comparison, we follow the protocol of [18] to split the datasets into training sets and test sets, where the split of training/test is 9843/2468 on ModelNet40 and 3991/908 on ModelNet10. The performance is evaluated in terms of accuracy over the corresponding test splits.

1) PERFORMANCE COMPARISON

We compare the proposed one to 21 existing 3D shape recognition methods, which includes 11 view-based methods [21]–[25], [27], [28], [53], [55], [56], [63], 5 point cloud-based methods [20], [43]–[45], [47], and 5 voxel-based methods [18], [19], [48], [49], [52]. These methods are the recent representative ones with published results on at least one test dataset; see Section II for a brief introduction to these methods. The results of all compared approaches are listed in Table 2, where methods of the same type are grouped together, and '-' denotes the unavailability of results.

Examining the results of view-based approaches in Table 2, we can find that on both datasets, our method outperforms

TABLE 2. Performance comparison of different approaches for 3D shape classification on ModelNet10 and ModelNet40 datasets in terms of classification accuracy (%).

Type	Method	ModelNet40	ModelNet10
Point cloud	SO-Net [47]	93.40	95.70
Point cloud	PointNet++ [44]	91.90	-
Point cloud	KCNet [45]	91.00	94.40
Point cloud	ECC [20]	83.20	90.00
Point cloud	PointNet [43]	89.20	-
Voxels	VRN-Ensemble [49]	95.50	97.10
Voxels	VRN [49]	91.30	93.60
Voxels	ORION [48]	-	93.8
Voxels	3D-GAN [52]	83.30	91.00
Voxels	VoxNet [19]	83.00	92.00
Voxels	3DShapeNets [18]	77.00	83.50
Multi view	RotationNet [22]	**97.37**	**98.46**
Multi view	MHBN (RGB+Depth) [24]	94.70	95.00
Multi view	MHBN (RGB) [24]	94.10	94.90
Multi view	Dominant (RGB+Depth+Surf) [25]	93.80	-
Multi view	Dominant (RGB) [25]	92.20	-
Multi view	SeqViews2SeqLabels [55]	93.40	94.82
Multi view	GVCNN [23]	93.10	
Multi view	MVCNN-MultiRes [63]	91.40	-
Multi view	PANORAMA-NN [28]	90.70	91.10
Multi view	Pairwise [53]	90.70	92.80
Multi view	MVCNN [21]	90.10	-
Multi view	GIFT [27]	83.10	92.35
Multi view	CNN + LSTM [56]	91.05	95.29
Multi view	Ours	94.65	95.26

other view-based methods except RotationNet [22], MHBN (RGB+Depth) [24] and CNN+LSTM [56]. Note that RotationNet uses 80 views captured from the vertices of a dodecahedron for recognition, which is about 6.7 times as ours. Thus, it is not surprising to see its superior performance. Note that a result of RotationNet using the same camera setup as ours is reported in [55], that is 90.65% on ModelNet40 and 93.84% on ModelNet10, which is below ours. Also note that compared to ours, MHBN (RGB+Depth) uses an additional depth features of objects. Even though the gap between MHBN (RGB+Depth) and ours is very minor (*i.e.* accuracy of 0.05%). When only using RGB data, MHBN yields lower classification accuracies than ours. With a similar structure to ours, CNN+LSTM [56] yielded comparable performance to ours on ModelNet10. However, our result is 3.6% higher than that of CNN+LSTM [56] on ModelNet40. The dominant-set clustering based approach [25] denoted by Dominant (RGB+Depth+Surf) also uses additional features including depth values and surface normals. Yet, it shows inferior performance to ours. In particular, we pay attention to MVCNN [21], MHBN (RGB) [24], SeqViews2SeqLabels [55], Dominant (RGB) [25] and GVCNN [23] which use similar frameworks to ours but with different aggregation modules. The effectiveness of the proposed LSTM-based aggregation module can be demonstrated by the performance improvement over these methods.

It can also be found from the results that both point cloud-based and voxel-based methods are generally not as good as view-based methods, except the VRN-Ensemble method [49] which outperforms ours on both datasets. However, as discussed by its authors [49], the power of this method comes from the ensemble with a mixture of predictions instead of the shape descriptor itself, which can be applied to any other methods including ours. When removing the ensemble module, the results of VRN [49] are worse than ours.

TABLE 3. Classification accuracy (%) on ModelNet40 by proposed method with different camera setups.

Number of Views (M)	Evaluation Angle (θ)	Accuracy (%)
8	30°	94.24
8	0°	92.93
12	30°	**94.64**
12	0°	92.24
24	30°	94.29

TABLE 4. Classification accuracy (%) on ModelNet40 with various-length input sequences on testing.

Number of Views	Accuracy(%)
12	94.64
6	94.04
3	92.42
1	90.84

Some of the misclassified samples are shown in Fig. 8. As we can see, since there are large ambiguities across categories, these objects are very challenging to classify even for human beings. For example, a flower in a pot can be classified not only as a plant but also as a flower pot.

2) BEHAVIOR ANALYSIS

In order to analyze how the multi-view representation influences the performances of our modules, we tested our method with different camera setups, including setting different numbers of views and different elevation angles of camera. Meanwhile, to figure out how well our method can handle a view sequence, we firstly train our method using full view representation and test its performance with partial view representation as input. Furthermore, to demonstrate the benefits brought from the bi-directional structure of LSTM, we also tested the performance of our method by replacing the bi-directional LSTM with a traditional one-way LSTM. The results with different camera setups and different LSTM structures are shown in Table 3 and Table 5. To save computational cost, in the tests of this subsection, we trained our network as follows. The VGG and the bi-LSTM networks were jointly trained with Back-Propagation over 100 epochs. The learning rate was initialized by 0.01 and decayed by 10 times every 80 epochs.

To evaluate the influence caused by the viewpoints of multi-view images, we train our networks using multi-view representation rendered under different camera steps. Results are listed in Table 3. We first check the influence by the elevation angle θ. It can be seen that the performance of our method drops moderately when changing the elevation angle

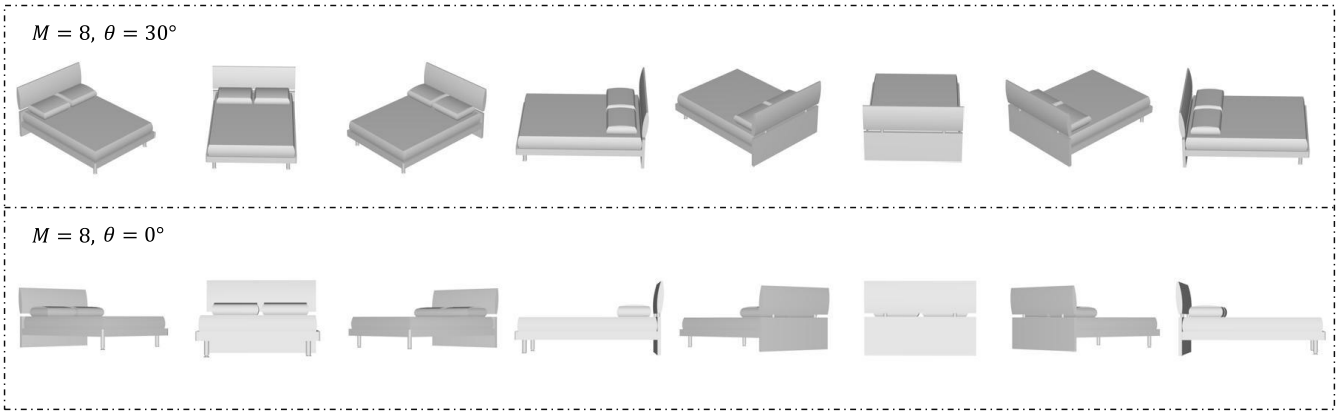


FIGURE 7. Examples of multi-view images of a 3D bed model captured by cameras elevated by 30° (top row) and 0° (bottom row) respectively.

TABLE 5. Classification accuracy (%) by proposed method with different structures of LSTM.

LSTM setup	ModelNet40	ModelNet10
one-way	93.80	94.82
two-way	**94.64**	**94.95**

TABLE 6. Classification accuracy (%) by models with different view aggregation techniques.

Aggregation Method	ModelNet40	ModelNet10
Max Pooling	93.44	94.16
Average Pooling	92.59	93.83
Sequence Aware Aggregation	**94.64**	**94.95**

of camera from 30° to 0°. This phenomenon is reasonable, as under the upright orientation assumption, the multi-view representation with $\theta = 30°$ contains more visual information than that $\theta = 0°$. An example on this issue is shown In Fig. 7, where we can see that setting $\theta = 30°$ can result in view images with more details on the object.

In comparison, the influence by the number of views is not as big as the elevation angle. Although using $M = 12$ views may obtain more information than using $M = 8$ views, the performance improvement of our method is very minor (less than 1%). This is probably that our LSTM-based module is very effective and efficient in exploiting the long-term dependencies of view sequences for 3D shape recognition, and $M = 8$ views are sufficient to generate informative and distinct shape descriptors. Furthermore, we also found in the experiments that our method can use fewer views to outperform many of the compared methods (*e.g.* [23]–[25]). In common-sense, using more views should help the network perform better because more views contain more information. However, as shown by the last line in Table 3, using too much views instead results in lower performance. Our conjecture is that the redundancy encoded in the overlong view sequences become a burden on training our network.

After training our network using multi-view representation with fixed viewpoint number, we then test its performance using less views, in order to see if it can still achieve promising results. The network using in Table 4 is trained on ModelNet40 under our default camera setup ($M = 12, \theta = 30°$). View sequences with views less than 12 are simply cut off from the full-length view sequences. Surprisingly, when fed with half-length view sequences ($M = 6$), the performance drop of our network is very minor. In this case, compared with the original full-length sequences, half of the static visual information is lost. This promising result demonstrates that the sequential information captured by our network can greatly help to relieve the loss of static visual information. However, when the view sequences are too short, their sequential information will be not enough to compensate the visual loss. So in Table 4, noticeable drop can be seen when the sequential length goes to only 3 or 1. But compared to other view-based approaches, even with extremely few views, our network performance is still competitive.

In addition, we analyze the influence of replacing the bi-directional LSTM with one-way LSTM in our method. From Table 5, noticeable improvements of using bi-directional LSTM over one-way LSTM are observed. Such improvements clearly demonstrate the benefits introducing the bi-directional structure to LSTM in our aggregation module.

In order to find out how well the sequence-aware view aggregation module can boost the network performance, we replaced the view aggregation module in our network with a max/average view-pooling layer. From Table 6 we can find that, the network with a max-pooling layer constantly outdoes the one with an average-pooling layer. It can also be seen that the sequence-aware module achieved the best results on both ModelNet40 and ModelNet10. On ModelNet10, the performance gain of the sequence-aware module is 0.79% over using the max-pooling. The performance gap is even higher on ModelNet40, which is 1.20%. Such results have demonstrated that the effectiveness of the

TABLE 7. Classification accuracy (%) by proposed method with different LSTM hidden state dimension.

Dimension	ModelNet40	ModelNet10
128 * 2	93.96	93.61
512 * 2	94.08	94.27
1024 * 2	94.41	94.82
2048 * 2	**94.64**	**94.95**

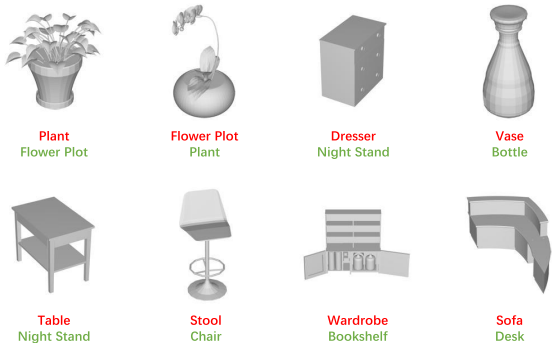


FIGURE 8. Some misclassified models. Predicted and ground-truth labels are highlighted in red and green respectively.

sequence-aware view aggregation module in exploiting the dependencies among views.

Finally, we study how the dimension of the LSTM hidden state influences network performances. Due to the use of bi-LSTMs, the dimension of an output state is twice that of a unit state. From Table 7, it is observed that the network performance increases as the dimension grows. Setting the unit hidden states to 2048 leads to the best result.The reason is probably that the vectors with higher dimension have the ability to encode more information. Moreover, this setting keeps the dimension of an LSTM output state consistent with that of a VGG feature.

B. 3D SHAPE RETRIEVAL

1) RETRIEVAL ON MODELNET40/10

On the ModelNet40 and ModelNet10 datasets, we use the same protocol as that in classification to split the training sets and test sets. After our network is trained, the 4096-dimensional feature from each bi-directional LSTM is extracted and all of these features are averaged as a single 4096-dimensional shape descriptor for retrieval. We followed [18] to conduct the retrieval experiments. Taking each sample in the test set as a query, we compute a ranked list on the remaining test samples, with the descending order of similarity measured by the ℓ_2 distance between two shape descriptors. The ranked lists generated from all test samples were then used to calculate the mean average precision (mAP), which is the mean of the average precision scores for each query. Average precision (AP) is defined as the average of precision values each time a positive sample is returned for a query.

TABLE 8. Performance comparison of 3D shape retrieval methods on ModelNet10 and ModelNet40 in terms of mAP (%).

Method	ModelNet40	ModelNet10
GVCNN [23]	85.70	-
PANORAMA-NN [28]	83.05	87.40
GIFT [27]	81.94	91.12
MVCNN [21]	79.50	-
DeepPano [64]	76.81	84.18
3DShapeNets [18]	49.20	68.3
CNN + LSTM [56]	84.34	**93.19**
Siamese CNN-BiLSTM [57]	83.30	-
Ours	**89.11**	90.37

For comparison, we select the methods [18], [21], [23], [27], [28], [56], [57], [64], which have published retrieval results on the ModelNet benchmark. The mAPs by different methods are compared in Table 8. On ModelNet40, our method achieved the best result, outperforming almost all the other methods by large margins. On ModelNet10, our method outperformed other compared methods except GIFT [27] and CNN+LSTM [56]. Note that the result of GIFT is generated with an easier setting than ours, where only 100 randomly-selected samples per category, instead of all samples from the dataset, are used for retrieval. Even though, the performance gap between our method and GIFT is just noticeable (less than 0.01 mAP difference). When compared with [56], [57] which also adopt bi-LSTM to aggregate view features, our method still yielded competitive results. On ModelNet40, our method outperformed [56], [57] noticeably with performance gap of 2.82% and 4.77% respectively.

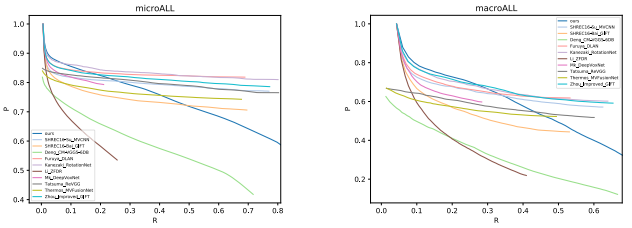
2) RETRIEVAL ON SHAPENETCORE55

ShapeNetCore55 is a subset of ShapeNet [65], which is the official dataset for retrieval competition SHREC2016 and SHREC2017 [62]. SHREC2017 provides two dataset versions: regular and perturbed. We only consider the regular version of this dataset. On this dataset, we follow the instruction of the SHREC2017 competition to construct the ranked lists on the test split and use the evaluation code provided by [62] for the test. The evaluation scheme on ShapeNetCore55 is different from those on ModelNet40 and ModelNet10 on two main points. First, the ranked list for a query contains the query itself. Second, at most 1000 entries are allowed per ranked list, *i.e.*, if a ranked list contains more than 1000 entries, the rest of the entries will be cut off.

Four metrics are used to evaluate the retrieval performances of the compared approaches, including PR (precision and recall), F-Score, mAP (mean average prediction), and NDCG (normalized discounted cumulative gain). The first three metrics are computed based on binary in-category versus out-of-category relevance, whereas the NDCG metric uses a graded relevance with grades 0, 1, 2, 3 that additionally considers the matching of sub-categories in ShapeNetCore55. Two schemes are used to average each of the

TABLE 9. Retrieval results on ShapeNetCore55.

Method	micro					macro				
	P@N	R@N	F1@N	mAP	NDCG@N	P@N	R@N	F1@N	mAP	NDCG@N
RotationNet [22]	0.810	0.801	**0.798**	0.772	0.865	0.602	0.639	**0.590**	0.583	0.656
Zhou_Improved_GIFT	0.786	0.773	0.767	0.722	0.827	0.592	0.654	0.581	0.575	0.657
Tatsuma_ReVGG	0.765	0.803	0.772	0.749	0.828	0.518	0.601	0.519	0.496	0.559
Furuya_DLAN	**0.818**	0.689	0.712	0.663	0.762	**0.618**	0.533	0.505	0.477	0.563
Thermos_MVFusionNet	0.743	0.677	0.692	0.622	0.732	0.523	0.494	0.484	0.418	0.502
Deng_CM-VGG5-6DB	0.418	0.717	0.479	0.540	0.654	0.122	0.667	0.166	0.339	0.404
Li_ZFDR	0.535	0.256	0.282	0.199	0.330	0.219	0.409	0.197	0.255	0.377
Mk_DeepVoxNet	0.793	0.211	0.253	0.192	0.277	0.598	0.283	0.258	0.232	0.337
MVCNN [21]	0.770	0.770	0.764	0.735	0.815	0.571	0.625	0.575	0.566	0.640
GIFT [27]	0.706	0.695	0.689	0.640	0.765	0.444	0.531	0.454	0.447	0.548
CNN + LSTM [56]	0.526	**0.899**	0.602	**0.810**	**0.868**	0.151	0.812	0.206	0.604	0.632
Ours	0.513	0.897	0.590	0.778	0.848	0.151	**0.881**	0.209	**0.634**	**0.677**



(a) microALL (b) macroALL

FIGURE 9. Precision-recall plots on ShapeNetCore55.

metrics on the samples, including the macro average which is the unweighted average over the entire dataset, and micro average that is computed across categories with adjustment for category sizes. Please refer to [62] for more details on the metrics.

In our experiments, the standard 70%/10%/20% training/validation/test split was used. We trained our network with the train split in 55 categories. After training, we applied the network to 3D shape retrieval on the test split. The shape descriptor of each query is extracted by averaging the 55-dimensional FC features from the network output.

We compare our method with all the participating methods in SHREC2017 [62] as well as the top two methods from the SHREC2016 iteration of the competition. Besides, we also cite the published results of [56] on ShapeNetCore55 for comparison. The evaluation results are shown in Table 9. It can be seen that, when only compared with the participating methods in SHREC2017/SHREC2016, our method achieves the best results in 5 out of the 10 evaluation metrics. It is worth mentioning that our method performed 0.6%/5.1% higher than the top method RotationNet in terms of micro/macro mAP. The MVCNN [21] achieved the rank-1 results in SHREC2016. In comparison, our method has a significant performance gain, demonstrating the effectiveness of our view aggregation module. The CNN+LSTM [56] behaves similarly to our method. Specifically, our method performs better in terms of the macro metrics whereas [56] is better under micro metrics. We noticed that the averaged precision values (P@N) of our method and [56] are unsatisfactory. The PR curves on ShapeNetCore55 are shown in Fig. 9. Note that the curves of [56] are not present in Fig. 9 as [56] has not published its PR results. In general, our method outperforms more than half of the compared methods, including the top method in SHREC2016 (*i.e.* GIFT [27]). It can be seen that RotationNet [22] achieved the best overall performance. In contrast, our method outperforms all the other methods at the beginning but the precision value drops a bit fast as the the recall value increases.

V. CONCLUSION

In this paper, we proposed an effective view-based approach for 3D shape recognition. Our approach is built upon a CNN-based view feature extraction module and a bi-directional LSTM-based view aggregation module. Benefiting from the capability of bi-directional LSTM in exploiting long-term dependencies of sequences, our view aggregation module can generate effective shape descriptors that encode the sequential semantic relations among views. Compared to existing view-based approaches that focus on the similarity of views, our method pays additional attention to the relations among views which are implicated in the dynamic process of view collection. When applied to 3D shape classification and retrieval, our method showed excellent performance and achieved state-of-the-art results.

In future, we would like to investigate the exploitation of other stochastic relations among views, as well as develop more advanced view aggregation techniques, to improve the performance of 3D shape recognition.

REFERENCES

[1] H. Liu, Y. Cong, C. Yang, and Y. Tang, "Efficient 3D object recognition via geometric information preservation," *Pattern Recognit.*, vol. 92, pp. 135–145, Aug. 2019.

[2] P. Fraga-Lamas, T. M. Fernández-Caramés, Ó. Blanco-Novoa, and M. A. Vilar-Montesinos, "A review on industrial augmented reality systems for the industry 4.0 shipyard," *IEEE Access*, vol. 6, pp. 13358–13375, 2018.

[3] Ó. Blanco-Novoa, T. M. Fernández-Caramés, P. Fraga-Lamas, and M. A. Vilar-Montesinos, "A practical evaluation of commercial industrial augmented reality systems in an industry 4.0 shipyard," *IEEE Access*, vol. 6, pp. 13358–13375, 2018.

[4] T. Kim, J. Kang, S. Lee, and A. C. Bovik, "Multimodal interactive continuous scoring of subjective 3D video quality of experience," *IEEE Trans. Multimedia*, vol. 16, no. 2, pp. 387–402, Feb. 2014.

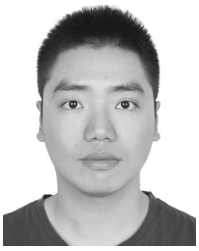
[5] Y. Sun, Z. Jiang, X. Qi, Y. Hu, B. Li, and J. Zhang, "Robot-assisted decompressive laminectomy planning based on 3D medical image," *IEEE Access*, vol. 6, pp. 22557–22569, 2018.
[6] M. Zhou, X. Hao, A. Eslami, K. Huang, C. Cai, C. P. Lohmann, N. Navab, A. Knoll, and M. A. Nasseri, "6DOF needle pose estimation for robot-assisted vitreoretinal surgery," *IEEE Access*, vol. 7, pp. 63113–63122, 2019.
[7] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 292–301.
[8] X. Sun, Y. Cai, S. Wang, X. Xu, and L. Chen, "Piecewise affine identification of tire longitudinal properties for autonomous driving control based on data-driven," *IEEE Access*, vol. 6, pp. 47424–47432, 2018.
[9] T. Li, M. Lee, C. Lin, G. Liou, and W. Chen, "Design of autonomous and manual driving system for 4WIS4WID vehicle," *IEEE Access*, vol. 4, pp. 2256–2271, 2016.
[10] H. W. Yu and B. H. Le, "A variational observation model of 3D object for probabilistic semantic SLAM," 2018, *arXiv:1809.05225*. [Online]. Available: https://arxiv.org/abs/1809.05225
[11] Y. Ma, Y. Wang, X. Mei, C. Liu, X. Dai, F. Fan, and J. Huang, "Visible/infrared combined 3D reconstruction scheme based on nonrigid registration of multi-modality images with mixed features," *IEEE Access*, vol. 7, pp. 19199–19211, 2019.
[12] P. Parkhiya, R. Khawad, J. K. Murthy, B. Bhowmick, and K. M. Krishna, "Constructing category-specific models for monocular object-SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2018, pp. 1–9.
[13] H. Pan and X. Yang, "Application of Internet of Things technology in 3D medical image model," *IEEE Access*, vol. 7, pp. 5508–5518, 2018.
[14] A. Polanczyk, M. Podgorski, M. Polanczyk, A. Piechota-Polanczyk, C. Neumayer, and L. Stefanczyk, "A novel patient-specific human cardiovascular system phantom (HCSP) for reconstructions of pulsatile blood hemodynamic inside abdominal aortic aneurysm," *IEEE Access*, vol. 6, pp. 61896–61903, 2018.
[15] Y. Gao, X. Xiang, N. Xiong, B. Huang, H. J. Lee, R. Alrifai, X. Jiang, and Z. Fang, "Human action monitoring for healthcare based on deep learning," *IEEE Access*, vol. 6, pp. 52277–52285, 2018.
[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, and A. C. Berg, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[18] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1912–1920.
[19] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Sep. 2015, pp. 922–928.
[20] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 3693–3702.
[21] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 945–953.
[22] A. Kanezaki, Y. Matsushita, and Y. Nishida, "RotationNet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5010–5019.
[23] Y. Feng, Z. Zhang, X. Zhao, R. Ji, and Y. Gao, "GVCNN: Group-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 264–272.
[24] T. Yu, J. Meng, and J. Yuan, "Multi-view harmonized bilinear network for 3D object recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 186–194.
[25] C. Wang, M. Pelillo, and K. Siddiqi, "Dominant set clustering and pooling for multi-view 3D object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2017, pp. 1–12.
[26] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, 1999, pp. 850–855.
[27] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. Jan Latecki, "Gift: A real-time and scalable 3D shape search engine," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5023–5032.
[28] K. Sfikas, T. Theoharis, and I. Pratikakis, "Exploiting the PANORAMA representation for convolutional neural network classification and retrieval," in *Proc. Eurograph. Workshop 3D Object Retr.*, vol. 8, 2017, pp. 1–7.
[29] C. Zhang and T. Chen, "Efficient feature extraction for 2D/3D objects in mesh representation," in *Proc. Int. Conf. Image Process.*, vol. 3, Oct. 2001, pp. 935–938.
[30] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3D shape descriptors," in *Proc. Symp. Geometry Process.*, vol. 6, 2003, pp. 156–164.
[31] R. M. Rustamov, "Laplace-Beltrami eigenfunctions for deformation invariant shape representation," in *Proc. EuroGraph. Symp. Geometry Process.*, 2007, pp. 225–233.
[32] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3D scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 5, pp. 433–449, May 1999.
[33] S. Belongie, J. Malik, and J. Puzicha, "Shape context: A new descriptor for shape matching and object recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 831–837.
[34] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud, "Surface feature detection and description with applications to mesh matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 373–380.
[35] M. Reuter, S. Biasotti, D. Giorgi, G. Patanè, and M. Spagnuolo, "Discrete Laplace–Beltrami operators for shape analysis and segmentation," *Comput. Graph.*, vol. 33, no. 3, pp. 381–390, Jun. 2009.
[36] J. Sun, M. Ovsjanikov, and L. Guibas, "A concise and provably informative multi-scale signature based on heat diffusion," *Comput. Graph. Forum*, vol. 28, no. 5, pp. 1383–1392, 2009.
[37] M. M. Bronstein and I. Kokkinos, "Scale-invariant heat kernel signatures for non-rigid shape recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1704–1711.
[38] L. Yi, H. Su, X. Guo, and L. J. Guibas, "SyncSpecCNN: Synchronized spectral CNN for 3D shape segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 6584–6592.
[39] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein, "Learning shape correspondence with anisotropic convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3189–3197.
[40] J. Xie, Y. Fang, F. Zhu, and E. Wong, "DeepShape: Deep learned shape descriptor for 3d shape matching and retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1275–1283.
[41] V. Hegde and R. Zadeh, "FusionNet: 3D object classification using multiple data representations," 2016, *arXiv:1607.05695*. [Online]. Available: https://arxiv.org/abs/1607.05695
[42] Z. Zhang, H. Lin, X. Zhao, R. Ji, and Y. Gao, "Inductive multi-hypergraph learning and its application on view-based 3D object classification," *IEEE Trans. Image Process.*, vol. 27, no. 12, pp. 5957–5968, Aug. 2018.
[43] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, no. 2, Jul. 2017, pp. 1–4.
[44] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Ann. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.
[45] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 4, Jun. 2018, pp. 4548–4557.
[46] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3D point cloud models," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 863–872.
[47] J. Li, B. M. Chen, and G. H. Lee, "So-net: Self-organizing network for point cloud analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9397–9406.
[48] N. Sedaghat, M. Zolfaghari, E. Amiri, and T. Brox, "Orientation-boosted voxel nets for 3D object recognition," in *Proc. Brit. Mach. Vis. Conf.*, 2017. [Online]. Available: http://lmb.informatik.uni-freiburg.de/Publications/2017/SZB17a
[49] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "Generative and discriminative voxel modeling with convolutional neural networks," 2016, *arXiv:1608.04236*. [Online]. Available: https://arxiv.org/abs/1608.04236
[50] G. Riegler, A. O. Ulusoy, and A. Geiger, "OctNet: Learning deep 3D representations at high resolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 3, Jul. 2017, pp. 3577–3586.

[51] H. Yu and B. H. Lee, "A variational feature encoding method of 3D object for probabilistic semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2018, pp. 3605–3612.
[52] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Proc. Ann. Conf. Adv. Neural Inf. Process. Syst.*, 2016, pp. 82–90.
[53] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, May 2016, pp. 3813–3822.
[54] K. Sfikas, I. Pratikakis, and T. Theoharis, "Ensemble of panorama-based convolutional neural networks for 3D model classification and retrieval," *Comput. Graph.*, vol. 71, pp. 208–218, Apr. 2018.
[55] Z. Han, M. Shang, Z. Liu, C.-M. Vong, Y.-S. Liu, M. Zwicker, J. Han, and C. P. Chen, "SeqViews2SeqLabels: Learning 3D global features via aggregating sequential views by rnn with attention," *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 658–672, Feb. 2018.
[56] C. Ma, Y. Guo, J. Yang, and W. An, "Learning multi-view representation with LSTM for 3-D shape recognition and retrieval," *IEEE Trans. Multimedia*, vol. 21, no. 5, pp. 1169–1182, Oct. 2018.
[57] G. Dai, J. Xie, and Y. Fang, "Siamese CNN-BiLSTM architecture for 3D shape representation learning," in *Proc. IJCAI*, 2018, pp. 670–676.
[58] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Trans. Signal Process.*, vol. 45, no. 11, pp. 2673–2681, Nov. 1997.
[59] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1961–1970.
[60] G. Parascandolo, H. Huttunen, and T. Virtanen, "Recurrent neural networks for polyphonic sound event detection in real life recordings," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2016, pp. 6440–6444.
[61] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc Int. Conf. Artif. Intell. Stat.*, 2010, pp. 249–256.
[62] M. Savva *et al.*, "Large-scale 3D shape retrieval from ShapeNet Core55: SHREC'17 track," in *Proc. Workshop 3D Object Retr.*, 2017, pp. 39–50. [Online]. Available: https://shapenet.cs.stanford.edu/shrec17/
[63] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 5648–5656.
[64] B. Shi, S. Bai, Z. Zhou, and X. Bai, "DeepPano: Deep panoramic representation for 3-D shape recognition," *IEEE Signal Process. Lett.*, vol. 22, no. 12, pp. 2339–2343, Dec. 2015.
[65] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, and H. Su, "ShapeNet: An information-rich 3D model repository," 2015, *arXiv:1512.03012*. [Online]. Available: https://arxiv.org/abs/1512.03012

YONG XU received the B.S., M.S., and Ph.D. degrees in mathematics from Nanjing University, Nanjing, China, in 1993, 1996, and 1999, respectively.

He was a Postdoctoral Research Fellow of computer science with the South China University of Technology, Guangzhou, China, from 1999 to 2001, where he became a Faculty Member and is currently a Professor with the School of Computer Science and Engineering. He is also a member of the Peng Cheng Laboratory and the Communication and Computer Network Laboratory of Guangdong. His current research interests include computer vision, pattern recognition, image processing, and big data.

CHAODA ZHENG received the B.Eng. degree in computer science from the South China University of Technology, in 2017. He is currently pursuing the Ph.D. degree. His research interests include computer vision, 3D data processing, and object recognition.



RUOTAO XU received the B.Eng. degree in computer science from the South China University of Technology, in 2015. He is currently pursuing the Ph.D. degree. His research interests include computer vision, image processing, and sparse coding.

YUHUI QUAN received the Ph.D. degree in computer science from the South China University of Technology, in 2013.

He was a Postdoctoral Research Fellow in mathematics with the National University of Singapore, from 2013 to 2016. He is currently an Associate Professor with the School of Computer Science and Engineering, South China University of Technology. He is also a member of the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information. His research interests include computer vision, image processing, and sparse representation.