Modeling 2D Appearance Evolution for 3D Object Categorization

Hasan F. M. Zaki[1,3], Faisal Shafait[2], Ajmal Mian[1]
[1]Computer Science and Software Engineering
The University of Western Australia, Crawley, WA, Australia
[2]National University of Sciences and Technology, Islamabad, Pakistan
[3]Mechatronics Engineering, International Islamic University Malaysia, Malaysia
hasan.mohdzaki@research.uwa.edu.au
faisal.shafait@seecs.edu.pk
ajmal.mian@uwa.edu.au

***Abstract*—3D object categorization is a non-trivial task in computer vision encompassing many real-world applications. We pose the problem of categorizing 3D polygon meshes as learning appearance evolution from multi-view 2D images. Given a corpus of 3D polygon meshes, we first render the corresponding RGB and depth images from multiple viewpoints on a uniform sphere. Using rank pooling, we propose two methods to learn the appearance evolution of the 2D views. Firstly, we train view-invariant models based on a deep convolutional neural network (CNN) using the rendered RGB-D images and learn to rank the first fully connected layer activations and, therefore, capture the evolution of these extracted features. The parameters learned during this process are used as the 3D shape representations. In the second method, we learn the aggregation of the views from the outset by employing the ranking machine to the rendered RGB-D images directly, which produces aggregated 2D images which we term as "3D shape images". We then learn CNN models on this novel shape representation for both RGB and depth which encode salient geometrical structure of the polygon. Experiments on the ModelNet40 and ModelNet10 datasets show that the proposed method consistently outperforms existing state-of-the-art algorithms in 3D shape recognition.**

I. Introduction

Since real-world environments consist of multi-dimensional vision cues, 3D shape representation has emerged as one of the most promising methods for object recognition. While object recognition in conventional 2D images has achieved remarkable success in the recent years, especially due to the discriminative power of the convolutional neural networks (CNN) and large-scale training datasets, the progress in 3D shape recognition has been limited. Most of the 3D shape recognition algorithms focus on the keypoint-based object labeling and instance matching [1], [2]. As many applications of 3D object recognition are emerging such as mobile robot navigation using low-cost depth sensors, expressive 3D feature representation serves as a key ingredient for building accurate and robust object recognition systems.

One of the fundamental questions that needs to be addressed in 3D object recognition is the design of a data representation that best captures the salient information of the 3D object shape. The natural and direct answer to this would be to design the representation based on the volumetric input of the 3D objects which has been used in recent works [3], [4], [5]. However, designing a representation especially using deep learning based techniques directly on the volumetric data is immensely expensive. Therefore, the resolution of the raw volumetric input needs to be significantly reduced which leads to a sub-optimal representation due to the loss of information. Moreover, as the training samples of these volumetric data are limited in numbers and manually annotating the training samples is cumbersome, thus the powerful deep networks such as CNN cannot be readily exploited to perform 3D object classification. In contrast, recent works have shown that building 3D shape classifiers does not require learning the representation explicitly from the volumetric data, but can be effectively achieved using their 2D image renderings [6], [7], [8].

Using a similar paradigm, we pose the problem of designing 3D object representation as learning the appearance evolution of the 3D shape based on the 2D image views. Concretely, we render each 3D polygon mesh into pairwise RGB and depth images from multiple viewpoints on a sphere. Based on these image sequences, we propose two methods to construct the 3D representation using a rank pooling algorithm [9]. In the first method, we train a view-invariant CNN model separately for RGB and depth image sequences. We then feed-forward each RGB-D image through these CNN models and extract the first fully connected layer activations. For each 3D polygon mesh, we learn the evolution of the 2D views by learning to rank these activations based on their indices and the feature representation of a 3D object is defined as the parameters learned by the rank pooling algorithm.

In the second method, we apply the ranking machine on the image sequence from the outset which outputs aggregated RGB and depth images which we call as "3D shape images". We then optimize new CNN models on these novel shape images for the 3D object classification task. The feature representation of each 3D polygon mesh is defined as the first fully connected layer activations of the CNN models. As the algorithms in both methods are applied directly on each instance and do not require dictionary construction from the training data, the techniques acquire fast computation and

978-1-5090-2896-2/16/$31.00 ©2016 IEEE

are scalable to any size of the testing data. We evaluate the proposed method on the benchmark ModelNet10 and ModelNet40 datasets and demonstrate that the proposed method consistently outperforms state-of-the-art algorithms.

II. Related Work

There is a large body of work on 3D object recognition that particularly emphasizes on the design of shape descriptors. Most of the methods utilize the native 3D representation as an input such as point cloud [3], [5], 3D occupancy grid [3] and 3D voxelized grid [4]. Earlier works in this context largely depend on the combinations of hand-crafted feature descriptors and classifiers [10], [11], [12]. As commonplace in other application domains, recent works shifted the research focus on applying deep learning based networks on the volumetric shape of the 3D objects directly to leverage the representational power of these networks. However, one apparent nuisance from such techniques is that, the original resolution of the 3D representation must be considerably reduced to allow the feasibility of network training which is otherwise intractable and prohibitively expensive to train. For instance, Wu *et al*. [4] train a convolutional deep belief network on a discretized 3D voxel grid of $30 \times 30 \times 30$. Similarly, Maturana and Scherer [3] train a GPU-based 3D CNN end-to-end using a volumetric occupancy grid computed from the point cloud data with a low resolution (*i.e*. $32 \times 32 \times 32$) as the input to the network. In contrast, we seek to maximally retain 3D information without having the need to compress the original data. Additionally, these methods cannot leverage massive amount of training data which is essential for effectively training data-hungry algorithms [13] as the samples for 3D models are much more restricted compared to the 2D images.

Our work is also related to the multi-view based recognition where a single 3D shape is decomposed into a 2D image sequence [6], [7]. Su *et al*. [6] proposed a CNN architecture that combines information from multiple views of a 3D object and outputs a compact aggregated 3D descriptor used for classification. Similarly, Johns *et al*. [7] further decomposed the images in an image sequence into RGB and depth images. Then, a simple two-stream CNN is used to classify each image pair independently. In this paper, we go beyond simple aggregation of the 2D views in order to get the 3D representation. We propose a structured method to model the appearance evolution of the multi-view images by learning to rank the image sequence and use the parameters learned during the modeling process as the 3D descriptors.

III. Proposed Methodology

In this section, we present our proposed method to encode 3D shape representation by modeling the appearance evolution of the 2D views. We first discuss about generating the RGB-D image sequences from a corpus of 3D polygon meshes. Next, we give details on the rank pooling algorithm used to model the appearance evolution. This technique is then applied in two different settings in order to encode the 3D shape representation used for classification.

A. Generating multi-view RGB-D data

Suppose we have 3D computer-aided design (CAD) based data similar to those in public 3D object repositories such as 3D Warehouse [1], Shapeways [2] and TurboSquid [3]. To this end, we use the 3D shape dataset of ModelNet [4] (details to follow in Sec. IV) to train the models and encode the 3D descriptors. To generate the multi-view RGB-D images, we first place the centroid of the target 3D polygon mesh at the centre of an enclosed sphere where several virtual cameras are employed in different positions. With the assumption that the object is always in the upright position against the gravity direction, we fix the virtual cameras at three different elevation angles namely at 30°, 0° and -30° and all cameras are pointing towards the object's centroid. At each elevation angle, twelve virtual cameras are positioned along the circle with a consistent separation of 30° between the two neighbouring cameras. Then, we render RGB and depth images at each camera position for dual-modality imagery. Therefore, for each 3D polygon mesh, we render $12 \times 3 = 36$ views for each RGB and depth sequence.

Following the method of Su *et al*. [6], object meshes are rendered under a perspective projection. For the rendering of RGB sequences, the reflection model of Phong [14] is used and a random color palette (since ModelNet does not have texture) is applied to each rendered view. Similarly, depth sequences are produced by taking the distance between the current camera lens and the facets of the polygon meshes. Although the number of viewpoints can be arbitrary using this method, we choose the most representative views such that it can replicate typical vision of a human or a mobile robot for object recognition.

Given the rendered RGB-D image sequence for each 3D object, one can simply treat the image sequence as an image set and apply any existing image set algorithm (*e.g*. [15]) to classify each image. However, the data distribution in an image set does not assume temporal consistency and coherence amongst the image sequence, albeit considering large variations amongst the image sequence. On the contrary, our rendered RGB-D sequence inherit strong temporal consistency as a result of consistent virtual camera trajectories. Therefore, we would want to exploit this important information for the encoding of our 3D shape representation. Particularly, we would want to capture the pattern that reveals how the multi-view appearance evolves in the sequence. Therefore, we propose two methods to model this evolution which will be further elaborated in the next sections.

B. Learning a view-invariant model and modelling the appearance evolution

Here, we discuss the **first method**, 2D Evolution Model (2DEM) to model the appearance evolution of the multi-view images. In this method, we first train a view-invariant

[1]3D Warehouse: https://3dwarehouse.sketchup.com/
[2]shapeways: http://www.shapeways.com/
[3]TurboSquid: http://www.turbosquid.com/

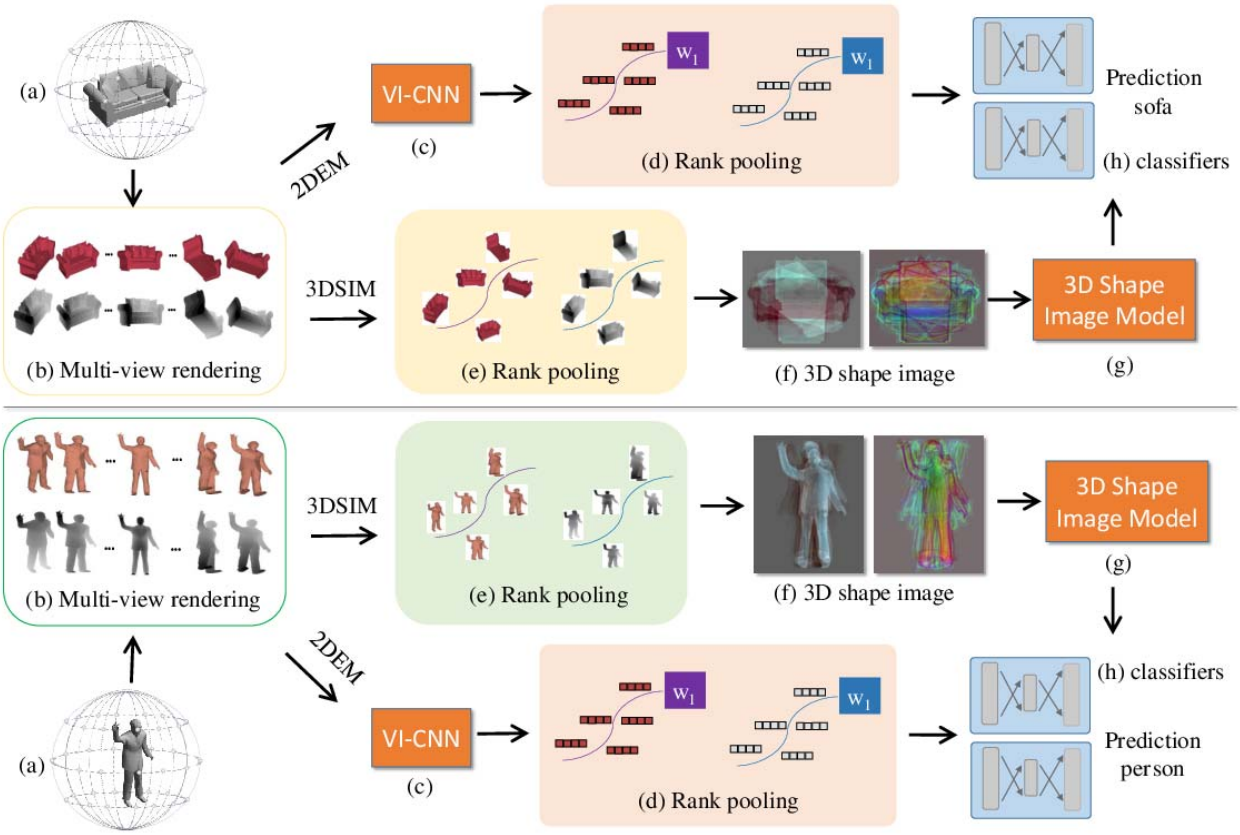

Fig. 1. Overview of the proposed methods. In this figure, we show the examples of *sofa* and *person* to represent the rigid and non-rigid objects in ModelNet40 dataset respectively. Starting from the (a) input of 3D polygon meshes, we render (b) multi-view pairwise RGB-D images from different elevation and azimuthal angles. Then, we propose two methods to encode the 3D shape representation based on these rendered 2D views. In the first method (2D Evo 1), we train a view-invariant CNN model (VI-CNN) and extract the first fully connected layer activations fc_6 as the features for each image. The evolution of these views according to the camera trajectory is then modelled using a linear ranking machine in (d). We propose to use the parameters learned by the ranking machine as the new feature representation for each 3D shape before predicting the object class. In the second method (2D Evo 2), we directly apply the learning to rank algorithm on the rendered images which produces a set of aggregated images termed as "3D shape images". Based on these novel images, we train another 3D Shape Image Model and extract the fc_6 activations as the feature representation for classification. Note that in our experiments, we consider RGB and depth separately hence the learning and encoding is done on each channel specific space.

CNN (VI-CNN) model separately for RGB and depth sequence using the rendered RGB-D images. For that purpose, we use the CNN model of VGG-f [16] which has been pre-trained on ImageNet [17] dataset as the initialization of our model parameters. Let us assume a CNN model which consists of consecutive modules of convolutional layer $L(k, f, s, p)$, max-pooling $MP(k, s)$, Local Contrast Normalization LCN, fully connected layers $FC(n)$ and rectified linear unit (ReLU) RL, where $k \times k$ is the receptive field size, f is the number of filters, s denotes the stride of the convolution and p indicates the spatial padding. The architecture of the model is given by: $L(11, 64, 4, 0) \rightarrow RL \rightarrow LCN \rightarrow MP(3, 2) \rightarrow L(5, 256, 1, 2) \rightarrow RL \rightarrow LCN \rightarrow MP(3, 2) \rightarrow L(3, 256, 1, 1) \rightarrow RL \rightarrow L(3, 256, 1, 1) \rightarrow RL \rightarrow L(3, 256, 1, 1) \rightarrow RL \rightarrow MP(3, 2) \rightarrow FC(4096) \rightarrow RL \rightarrow FC(4096) \rightarrow RL \rightarrow FC(1000)$. We refer to the fully connected layers as fc_6, fc_7 and fc_8 respectively.

The learning process of our VI-CNN is performed by replacing the last fully connected layer with a new randomly generated vector of dimension C, where $C = 40$ is the number of categories in ModelNet40 dataset. Moreover, we also add a dropout layer [18] in between fully connected layers towards the end of the network to prevent the neural networks from overfitting. Finally, we add a softmax loss layer at the top of the network and train the network using standard stochastic gradient descent with back-propagation. We use a relatively small global learning rate ($1\text{e}{-5}$) while we multiply the learning rate by ten at the newly added layer. As suggested by Chatfield *et al.* [16], the momentum is set to 0.9 and the weight decay is 0.0005. We train the entire network for multiple epochs and stop the learning process when the validation curve stabilizes into convergence.

After the completion of the VI-CNN learning, we then feed-forward all the rendered RGB and depth images through VI-CNN for RGB and depth respectively and extract the 4096-dimensional fc_6 activations as the features for each image.

Formally, let a rendered image be represented by a vector $\mathbf{x} \in \mathbb{R}^d$ where $d = 224 \times 224 \times 3$ is the default image dimension for the VGG-f model. Then, a 3D polygon mesh composes of a sequence of such rendered images, $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$. The feed-forward through VI-CNN produces a mapping from $\mathbf{X} \in \mathbb{R}^d \rightarrow \mathbf{V} \in \mathbb{R}^D$, where $D = 4096$ is the dimension of each mapped vector $\mathbf{v}_i$. Our goal is to model the evolution of these vectors $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N]$, where $N = 36$ is the number of viewpoints.

As the rendering of the 3D meshes follows the same camera trajectories, it is intuitive that these trajectories capture the salient pattern of the evolving shape appearance. Therefore, we propose to put a constraint into the ordering of the views such that vectors with bigger indices will always precede those with the smaller indices *i.e.* $\mathbf{v}_1 \prec \cdots \prec \mathbf{v}_i \prec \cdots \prec \mathbf{v}_N$. This relative ordering can be modeled using a ranking machine [9], [19]. Concretely, let $\mathbf{v}_{ia}$ and $\mathbf{v}_{ib}$ be pairwise vectors in the sequence, thus a ranking machine learns to optimize a linear function $\Psi(\mathbf{v}; \mathbf{w})$ parametrized by the weight vector $\mathbf{w}$, such that $\forall i_a, i_b,\ \mathbf{v}_{ia} \prec \mathbf{v}_{ib} \iff \mathbf{w}^T.\ \mathbf{v}_{ia} < \mathbf{w}^T.\ \mathbf{v}_{ib}$. On the basis of RankSVM [20] for structural risk minimization and max-margin formulation, the objective function can be expressed as

$$\begin{aligned} &\arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{\forall a,b,\mathbf{v}_{ia} \prec \mathbf{v}_{ib}} \epsilon_{ab} \\ &s.t.\ \mathbf{w}^T.(\mathbf{v}_{ib} - \mathbf{v}_{ia}) \geq 1 - \epsilon_{ab} \\ &\epsilon_{ab} \geq 0. \end{aligned} \tag{1}$$

The ranking function Ψ learns to sort the indices of the rendered image sequence. As the parametrization is characterized by the weight $\mathbf{w}$, the trajectories which encode the appearance evolution of the views can be modelled by this parameter. Therefore, we propose to use the parameter $\mathbf{w}$ as the feature representation for each 3D shape. Using the ranking machine to encode the 3D shape representation poses several advantages. Firstly, the method is dictionary free which is in contrast to typical learning based algorithms which need to explicitly learn a dictionary from a set of training images. As such, the method we used is computationally efficient and scalable to larger testing data. Moreover, the parameters learned using the ranking function lie in the same feature space as the rendered sequence vectors $\mathbf{V}$ [9]. Although the ranking method models the evolution of the appearance in a linear space, our VI-CNN has already encoded complex non-linearity in the 2D sequence which is important to model distributions of high variability.

C. 3D Shape Image Model

2D Evolution Model (2DEM) encodes the non-linear features of the sequence followed by a linear appearance evolution modelling. In the **second method**, 3D Shape Image Model (3DSIM), we alter the pipeline by first modeling the appearance evolution on the sequence input followed by encoding of the non-linearities via CNN training. Particularly, instead of applying the ranking machines on $\mathbf{V}$, we now apply the same algorithm directly on the image sequence *i.e.* the pixels of the rendered RGB and depth images. As mentioned earlier, the ranking function learns the parameters $\mathbf{w}$ with the same space and dimension as the input, thus $\mathbf{w}$ will be a real vector which can be interpreted as an aggregated image of the RGB and depth sequence respectively. We term this aggregated image as "3D shape image" as it represents the 3D shape in terms of its structurally aggregated 2D appearance. Fig. 2 shows some samples of the 3D shape images.

Then, we learn a 3D Shape Image model by fine-tuning the VGG-f model using these 3D shape images as inputs to the network with the same architecture and settings as described in Sec. III-B. We train the CNN separately for RGB and depth sequences. We proceed the learning up to 100 epochs as the validation curve converges. Similar to the learning of VI-CNN, we augment the training data by horizontally mirroring the images with a probability of 0.5 and normalize them using the mean image of ImageNet dataset since the model used for initialization was trained on this dataset. After the learning, we feed-forward the 3D shape images through the learned model and extract the fc_6 activations as the feature representation for each 3D shape before classification.

D. Implementation and Classification

To learn the VI-CNN and the 3D Shape Image Model, we use the Matlab based Matconvnet [21] toolbox with Tesla K40 GPU to implement the CNN architecture and learning. As for the ranking machines, we employ the implementation of Fernando *et al*. [9] with Liblinear SVM [22] library for efficient RankSVM implementation. The depth images are encoded following the method of Zaki *et al*. [23], where the original depth image is concatenated with the magnitude of gradient and the direction of gradient maps which then produces a three-channel encoded depth image. Note that other plausible depth encoding techniques such as HHA [24] and depth-jet [25] can also be used to fit the typical input setting of pre-trained CNNs.

In all cases, we use the 4096-dimensional feature vector of fc_6 as representation of each rendered image and representation of each 3D shape in the first and second method, respectively. We perform dimensionality reduction by PCA which compresses the 4096-dimensional feature vectors into 1000 dimensions. For the classification, we employ the non-linear classifiers of Huang *et al*. [26] which has been shown to be efficient for object categorization [27].

IV. Experiments

We implemented and evaluated our proposed methods on the challenging ModelNet [4] which is a benchmark testbed for 3D shape classification. The dataset comprises of about 127,915 3D CAD meshes from 662 categories and growing [4]. In contrast to several 3D object datasets, this dataset contains the combination of rigid and non-rigid objects such as *airplane*, *person* and *car*. In this paper, we experimented with

[4]Available at http://3dshapenets.cs.princeton.edu/

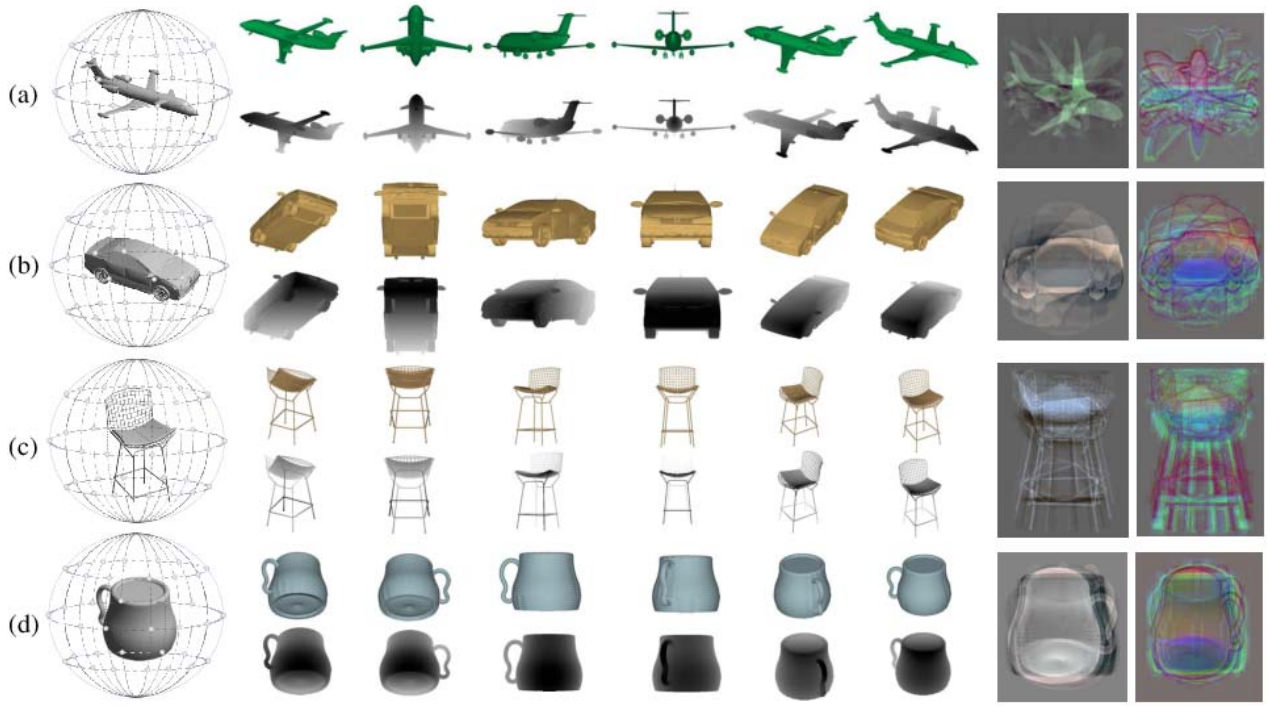

Fig. 2. Sample images of (a) airplane, (b) car, (c) stool and (d) cup in ModelNet40 [4] dataset. From left: the 3D polygon meshes enclosed in a uniform sphere (column 1), rendered RGB-D images with two samples from each elevation angle (column 2 to 7) and the 3D shape images as the output of applying the ranking machines on RGB (column 8) and depth sequence (column 9).

two subsets: ModelNet10 containing 10 shape categories and ModelNet40 which is organized into 40 categories. The procedure of Wu *et al*. [4] is adopted for evaluation. Specifically, the data in ModelNet40 were split into balanced training and testing sets where the first 80 meshes in each category's "train" folder (or all meshes for the case where the category has an insufficient number of meshes) are selected for training while the validation is conducted on the first 20 meshes in the "test" folder. The final dataset has 3183 shapes for training and 800 shapes for testing. Therefore, following the rendering method discussed in Sec. III-A, we rendered the 3D meshes into a total of 3983 $\times$ 36 = 143,388 multi-view RGB-D images.

A. Model Ablation

Firstly, we examine our methods, 2DEM as described in Sec. III-B and 3DSIM as described in Sec. III-C with different settings to analyse the contributions and effects of different elements in the methods. All results for model ablation are tabulated in Table I. We report the classification accuracy of using individually the rendered RGB images, depth images and the combination of both. The combination of both channel is done by concatenating the final channel-specific representations before classification. As for the 2DEM, we also report the accuracy recorded when different elevation angles are used during the rendering process. Note that the learning of VI-CNN model is done on the entire rendered images, whereas these different settings are done at feature extraction level before training and testing the classifiers.

We also include two baseline methods which are used to aggregate the multi-view descriptors V. These methods have been proven to be effective to aggregate multiple frames of videos and constitute state-of-the-art for the dynamic texture and scene classification task albeit the simplicity of the methods [28]. We briefly describe the methods here for completeness. The first baseline method is the first-order statistics which is simply the mean computed on the multi-view vectors $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N]$, given by $\mathbf{u} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{v}_i$. The second baseline method is the second-order statistics of the multi-view vectors $\mathbf{V}$ given by the covariance matrix $\mathbf{Y} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}_i - \mathbf{u})(\mathbf{v}_i - \mathbf{u})^T$. However, taking the entire covariance matrix $\mathbf{Y}$ would yield a $\frac{D(D+1)}{2}$-dimensional features since the matrix $\mathbf{Y}$ is symmetric which will lead to the curse of dimensionality. Thus, we only take the entries in the matrix diagonal as the feature representation which is in the same D = 4096 dimension. Intuitively, taking the mean of the multi-view features reflects the average behaviour of the views in the camera trajectory. On the other hand, the diagonal of $\mathbf{Y}$ denotes the relative variance and correlation of the views. As both methods ignore the exact indices of the rendered images, contrary to our methods, both can be seen as orderless based

TABLE I

PERFORMANCE COMPARISON IN TERMS OF CLASSIFICATION ACCURACY (%) OF THE PROPOSED METHODS (2DEM AND 3DSIM) WITH SEVERAL ORDERLESS BASED BASELINE METHODS FOR THE TASK OF 3D SHAPE CLASSIFICATION IN MODELNET10 AND MODELNET40 DATASETS.

			ModelNet10				**ModelNet40**			
Method	Remark	Elevation Angle	RGB	Depth	RGB-D	**Max. Acc.**	RGB	Depth	RGB-D	**Max. Acc.**
1st-order statistics	baseline	30°	89.5	91.5	91.5	91.5	86.5	88.0	88.1	88.1
		0°	82.5	89.0	90.5	90.5	83.3	86.0	88.1	88.1
		-30°	88.0	88.0	90.5	90.5	85.3	85.5	87.9	87.9
		all	87.5	90.5	90.5	90.5	86.9	88.0	88.1	88.1
2nd-order statistics	baseline	30°	88.5	90.5	90.0	90.5	82.6	80.1	81.6	82.6
		0°	85.5	88.5	87.5	88.5	80.1	79.6	82.4	82.4
		-30°	86.5	87.5	88.0	88.0	81.4	78.5	80.4	81.4
		all	88.1	89.0	90.0	90.0	86.8	87.8	88.1	88.1
2DEM	proposed	30°	**91.0**	89.0	91.5	91.5	86.3	84.1	85.9	86.3
		0°	88.5	88.5	89.0	89.0	86.1	83.1	86.0	86.1
		-30°	91.0	89.0	90.0	91.0	84.9	83.8	85.9	85.9
		all	89.5	**92.3**	**94.5**	**94.5**	**89.0**	**90.4**	**90.6**	**90.8**
3DSIM	proposed	all	87.0	87.1	88.0	88.0	85.1	86.3	86.8	86.8

encoding methods.

As can be seen, our proposed method 2DEM consistently gives the highest accuracy for every channel-specific and combined channels recognition while the performance of 3DSIM is at par with the baseline methods. Interestingly, almost in all cases, using only the images rendered at 30° elevation angle gives the best performance for all methods compared to the other angles which shows that the views captured from these viewpoints are more representative for 3D shapes than the other two angles. This is more apparent in ModelNet10 where categories such as *bed*, *toilet* and *bathtub* are better recognized from an elevated angle while the viewpoints at negative elevation angles do not well-characterize the objects. In the case of 2DEM, modelling the evolution from different elevation angles does not significantly effect the recognition accuracy. This might be due to the property of the algorithm that models the evolution according to the camera trajectories. As the trajectories are all in the ellipsoid shape, therefore the algorithm tends to model similar evolution features for all elevation angles. In contrast, modelling the evolution while considering all viewpoints in the sphere significantly boosts the recognition performance. This shows that modeling the evolution from a longer and varied camera trajectory produces a representation of higher discrimination for 3D shape recognition.

Furthermore, the recognition patterns of 2DEM and 3DSIM brings about interesting hypotheses. Firstly, the difference in the classification accuracy of both methods suggests that modeling the appearance evolution in a discriminative feature space (*i.e.* as CNN extracted features) is more effective than training a discriminative model on the evolution of the appearances (represented by the 3D shape images). Besides, the 2DEM method can directly benefit from the abundance of rendered training data while the training samples in 3DSIM are much more deficient. While this can be improved using data jittering technique [29], the high accuracy of the method could open up many possibilities for the 3D based recognition especially on representing 3D shapes by 2D aggregated images.

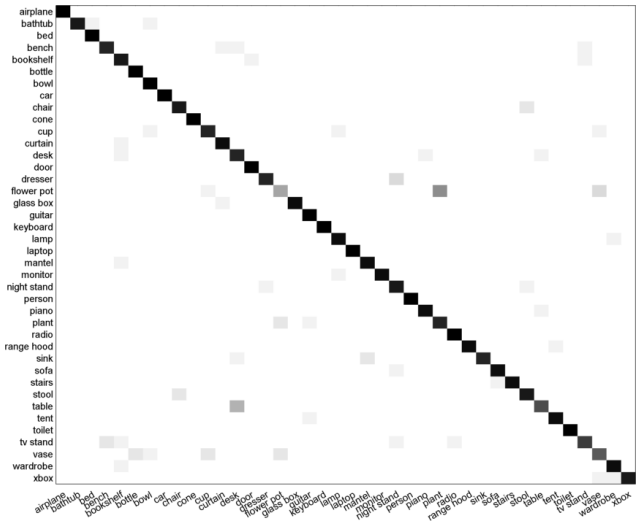

Fig. 3. Confusion matrix for the object categorization using our proposed 2DEM descriptor on the ModelNet40 [4] dataset. Note the strong diagonal entries of the matrix which reflects the high efficacy of the proposed method to discriminate between object classes. The figure is best viewed with magnification.

B. Comparative Analysis

Our proposed method is compared against various state-of-the-art algorithms for 3D shape classification. These include the techniques which used native 3D representation as input namely Spherical Harmonics descriptor (SPH) [30], LightField descriptor (LFD) [31], 3D Shapenets [4] and 3D CNN based VoxNet [3]. The 2D input based methods include Multi-view CNN (MVCNN) [6], deep panorama (DeepPano) [32], GPU acceleration and Inverted File Twice (GIFT) [8] and Pairwise descriptor (Pairwise) [7].

As depicted in Table II, 2D based methods outperform 3D based ones with a significant margin. Our method outperforms the deep learning based method with 3D input of 3D Shapenets and VoxNet by 10%. This is largely due to

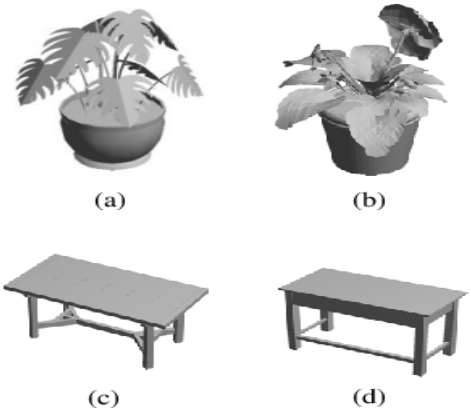

Fig. 4. Sample of off-diagonal entries of the confusion matrix in Fig. 3. (a) *plant* is misclassified as (b) *flower pot* and (c) *table* is misclassified as (d) *desk*.

TABLE II
PERFORMANCE COMPARISON IN TERMS OF CLASSIFICATION ACCURACY (%) OF THE PROPOSED METHOD AND STATE-OF-THE-ART METHODS FOR TASK OF 3D SHAPE CLASSIFICATION IN MODELNET10 AND MODELNET40 DATASETS.

Method	Remark	ModelNet10	ModelNet40	Input
SPH [30]	3D descriptor	79.8	68.2	3D
LFD [31]	3D descriptor	79.9	75.5	3D
3D Shapenets [4]	CVPR '15	83.5	77.3	3D
VoxNet [3]	IROS '15	92.0	83.0	3D
MVCNN [6]	ICCV '15	–	90.1	2D
DeepPano [32]	SPL '15	85.5	77.6	2D
GIFT [8]	CVPR '16	92.4	83.1	2D
Pairwise [7]	CVPR '16	91.3	89.2	2D
2DEM	this work	**94.5**	**90.8**	2D

the nature of the input where the compared methods used a limited number of training shapes as well as compressed voxels to train deep network from scratch while our method can directly utilize pre-trained CNN models to initialize our model. Our method also recorded a considerable improvement from the closest competitors, MVCNN and Pairwise, although the former method devised a specialized CNN architecture to aggregate the rendered 2D data while the latter aggregated the multi-modal information (grayscale and depth) explicitly in the learning of CNN. In contrast, our method constructs a more structured aggregation technique by learning to rank the views from more viewpoints (which means longer camera trajectory) which can capture how the appearance of the views evolve in the trajectory. The confusion matrix for the classification of ModelNet40 is depicted in Fig 3 and Fig. 4 shows sample misclassification, highlighting the low inter-class variation challenge in the dataset.

V. CONCLUSION

In this paper, we have presented methods to encode 3D shape representation based on the rendered 2D images. In the first method, we proposed to learn the order of the views representation and use the learned parameters as the 3D representation while in the second method, we constructed 3D shape images based on a ranking machine and learn a CNN model from the novel images. Experimental results demonstrate that our method significantly outperforms existing algorithms for the task of 3D object classification in ModelNet40 and ModelNet10 datasets.

ACKNOWLEDGMENT

Corresponding author is sponsored by the Ministry of Education Malaysia and this research was supported by the Australian Research Council (ARC) Discovery Project DP160101458. We thank NVIDIA for donating the Tesla K-40 GPU.

REFERENCES

[1] A. S. Mian, M. Bennamoun, and R. Owens, "Keypoint detection and local feature matching for textured 3D face recognition," *International Journal of Computer Vision*, vol. 79, no. 1, pp. 1–12, 2008.
[2] S. A. A. Shah, M. Bennamoun, and F. Boussaid, "A novel 3D vorticity based approach for automatic registration of low resolution range images," *Pattern Recognition*, vol. 48, no. 9, pp. 2859–2871, 2015.
[3] D. Maturana and S. Scherer, "Voxnet: A 3D convolutional neural network for real-time object recognition," in *Proc. IROS 2015*, pp. 922–928.
[4] 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. CVPR 2015*, pp. 1912–1920.
[5] K. Lai, L. Bo, and D. Fox, "Unsupervised feature learning for 3D scene labeling," in *Proc. ICRA 2014*, pp. 3050–3057.
[6] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. ICCV 2015*, pp. 945–953.
[7] E. Johns, S. Leutenegger, and A. J. Davison, "Pairwise decomposition of image sequences for active multi-view recognition," in *Proc. CVPR 2016*, June.
[8] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. Jan Latecki, "Gift: A real-time and scalable 3D shape search engine," in *Proc. CVPR 2016*, June.
[9] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.
[10] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *Proc. ECCV 2004*, pp. 224–237.
[11] J. Behley, V. Steinhage, and A. B. Cremers, "Performance of histogram descriptors for the classification of 3D laser range data in urban environments," in *Proc. ICRA 2012*, pp. 4391–4398.
[12] S. Song and J. Xiao, "Sliding shapes for 3D object detection in depth images," in *Proc. ECCV 2014*, pp. 634–651.
[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[14] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, pp. 311–317, 1975.
[15] M. Hayat, M. Bennamoun, and S. An, "Deep reconstruction models for image set classification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 4, pp. 713–727, 2015.
[16] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. BMVC*, 2014.
[17] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.
[18] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
[19] T.-Y. Liu, "Learning to rank for information retrieval," *Foundations and Trends in Information Retrieval*, vol. 3, no. 3, pp. 225–331, 2009.

[20] T. Joachims, "Training linear svms in linear time," in *Proc. of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006, pp. 217–226.

[21] A. Vedaldi and K. Lenc, "Matconvnet: Convolutional neural networks for matlab," in *Proc. of the 23rd ACM international conference on Multimedia*. ACM, 2015, pp. 689–692.

[22] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, pp. 1871–1874, 2008.

[23] H. F. M. Zaki, F. Shafait, and A. Mian, "Convolutional hypercube pyramid for accurate RGB-D object category and instance recognition," in *Proc. ICRA 2016*, pp. 1685–1692.

[24] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. ECCV 2014*, pp. 345–360.

[25] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition," in *Proc. IROS 2015*, pp. 681–687.

[26] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[27] H. F. Zaki, F. Shafait, and A. Mian, "Localized deep extreme learning machines for efficient RGB-D object recognition," in *Proc. DICTA*, 2015, pp. 1–8.

[28] X. Qi, C.-G. Li, G. Zhao, X. Hong, and M. Pietikäinen, "Dynamic texture and scene classification by transferring deep image features," *Neurocomputing*, vol. 171, pp. 1230–1241, 2016.

[29] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 806–813.

[30] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Rotation invariant spherical harmonic representation of 3 d shape descriptors," in *Symposium on geometry processing*, vol. 6, 2003, pp. 156–164.

[31] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3d model retrieval," in *Computer graphics forum*, vol. 22, no. 3. Wiley Online Library, 2003, pp. 223–232.

[32] B. Shi, S. Bai, Z. Zhou, and X. Bai, "Deeppano: Deep panoramic representation for 3-D shape recognition," *IEEE Signal Processing Letters*, vol. 22, no. 12, pp. 2339–2343, 2015.