

Received January 7, 2021, accepted January 26, 2021, date of publication February 3, 2021, date of current version February 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3056613

2DSlicesNet: A 2D Slice-Based Convolutional Neural Network for 3D Object Retrieval and Classification

ILYASS OUZZANI TAYBI¹, TAOUFIQ GADI¹, AND RACHID ALAOU^{2,3}

¹LIIMSC Laboratory, Faculty of Sciences and Techniques, Hassan First University, Settat 26000, Morocco

²LRIT Laboratory, Faculty of Sciences, Mohammed V University, Rabat 10000, Morocco

³LASTIMI Laboratory, Higher School of Technology—Sale, Mohammed V University, Rabat 10000, Morocco

Corresponding author: Ilyass Ouazzani Taybi (ilyass.ouazzani@gmail.com)


ABSTRACT 3D data can be instrumental to the computer vision field as it provides insightful information about the full 3D models' geometry. Recently, with easy access to both computational power and huge 3D databases, it is feasible to apply convolutional neural networks to automatically extract the 3D models' features. This paper presents a novel approach, called 2DSlicesNet, which deals with the issue of 3D model retrieval and classification using a 2D slice-based representation with a 3D convolutional neural network. The assumption in this context is that similar 3D models will be composed of almost identical 2D slices. Therefore, we first transform each normalized 3D model into a set of 2D slices corresponding to its first main axis, and then use them as input data to our 3D convolutional neural network. Experimental results and comparison with state-of-the-art approaches, using ModelNet10 and ModelNet40 datasets, prove that our proposed 2DSlicesNet approach can reach notable rates of accuracy in classification and retrieval.

INDEX TERMS Deep learning, 2D slices, 3D convolutional neural network, 3D object classification, 3D object retrieval.

I. INTRODUCTION

With the rapid advancement of 3D object capturing instruments and computing power, there is a growing number of 3D models in different areas [1], such as medical simulation, computer vision, computer graphics, computer-aided design and architectural design. As opposed to model recognition and retrieval on 2D data, classifying and retrieving models from 3D information is a progressively viable and sensible errand. In that capacity, addressing 3D model classification and retrieval is a pertinent research subject. As a result, it has lately drawn considerable attention from researchers [2].

Early works of 3D model retrieval and classification are generally founded on 3D objects, where low-level characteristic-based approaches [3], [4] and high-level structure-based approaches [5], [6] have been utilized. Recently, the effectiveness of deep learning approaches [7], particularly convolutional neural network (CNN), has speeded the development of 3D model recognition and retrieval, and have demonstrated their predominance compared to traditional approaches.

The associate editor coordinating the review of this manuscript and approving it for publication was Tyson Brooks .

Among the existing deep learning approaches, using 2D slices as the input data to a CNN has recently demonstrated to be an effective approach that has accomplished state of the art 3D model recognition in Gomez-Donoso *et al.* [8] paper. In fact, the authors proposed LonchaNet, a 3D model classification approach based on 2D slices. They extracted three 2D slices for each 3D model corresponding to XY, XZ, and YZ planes. After, they used the drawn 2D slices as the input to three independent GoogLeNet networks for learning and extracting 2D slices features, which are joined in a layer prior to the classification layer. The authors tested their method in the ModelNet10, which is both a 3D database and a challenge. The suggested approach obtained a 94.37% classification accuracy, outperforming most existing methods that participated in the challenge. However, the LonchaNet approach extracts only one 2D slice for each 3D object's axis, which can, sometimes, lead to 2D slices being completely different for some 3D models of the same category, or similar 2D slices for 3D models of distinct classes. In addition, to pull out the 3D model's slices, the authors make cross-sections with a width of 5% of the 3D model size, and then they project the points that fall inside these sections in their planes, which could cause loss of information due to the projection.

To address all these limitations, a novel approach called 2DSlicesNet is posited, which combines successive 2D slices for 3D characteristic learning by using 3D convolutional neural network (3DCNN). In our approach, we produce 2D slices directly from the intersection of each normalized 3D model with 64 planes equally spaced and orthogonal to the Ox axis, which aims to avert the 2D slice loss of information due to projection. We also represent each 3D model by a set of successive 2D slices. In this way, we assure that similar 3D models will be represented by almost identical 2D slices, and the order of 2D slices is preserved by using 3DCNN. Indeed, to raise the learned features' discriminability, 2DSlicesNet aggregates not only the information within all 2D slices but also the successive spatiality amongst the 2D slices, which makes 2DSlicesNet learn the complete features in the set of 2D slices of each object.

To assess the performance of the proposed 2DSlicesNet system, we have conducted experiments on ModelNet10 and ModelNet40, which are both 3D databases and challenges. The findings of our experiments demonstrate that our proposed approach can generate comparable results, on both 3D model retrieval and classification tasks, in comparison with the state-of-the-art methods, which show the efficiency of the 2DSlicesNet.

The remainder of the paper is organized as follows. In the second section, we shortly review some of the previous works on 3D object recognition and retrieval. In the third section, we briefly discuss Convolutional Neural Network. Then, in the fourth section, we introduce our approach, which is entitled 2DSlicesNet. In the fifth part, we present and analyze the results of our experiments and draw conclusions. Finally, in the last part, we conclude this paper.

II. RELATED WORK

In recent years, 3D object retrieval and classification have been probed. In this section, we briefly shed light on notable handcrafted feature-based approaches and deep learning-based approaches.

A. HAND-CRAFTED FEATURE-BASED APPROACHES

The traditional hand-crafted feature-based approaches can be broadly divided into two groups [9], [10]: 2D image-based approaches [11], [12] [13], [14] and model-based approaches [15], [16] [17].

2D image-based approaches use a set of images to describe the 3D object. Light Field Descriptor (LFD) [18], the first typical 3D object image-based descriptor, 3D objects are represented by a set of ten images captured from the vertices of a dodecahedron over a hemisphere. In [19], the probabilistic corresponding was used to measure the resemblance between 3D objects. In [20], the authors generate an ensemble of 2D slices corresponding to specific axes. The Apriori algorithm was used to pick out the most distinctive ones. In [21], a set of panoramic images were extracted from the 3D object to represent the surface of model and the orientation.

One of the most commonly model-based approaches is the statistical approach, which can be employed to characterize the attributes' distributions. In [22], the authors represent the 3D object by a 3D closed curve, which is used to yield feature vectors. They combine two novel descriptors; the dot product and the area descriptors that typify the reconstructed 3D closed curve to portray the 3D curve analysis descriptor. In [3], the authors used the shape distribution to compute the similarity based on angle, distance, volume, and area between random surface points. In [23], the Spherical Harmonic (SPH) Representation is introduced, which is a rotation invariant representation of spherical functions in terms of the energies at different frequencies. The proposed method is the Gaussian Euclidean Distance Transform's volumetric representation of a 3D model, symbolized by spherical harmonic frequencies' norms. In [24] the Multi-Fourier Spectra approach was proposed by increasing the feature vector with spectral clustering. The proposed descriptor was composed of four separate Fourier spectra with periphery enhancement. It was capable of capturing the 3D object's intrinsic characteristics regardless of the 3D object's position, orientation, and scale. For 3D shape retrieval task, in [25] a probabilistic generative signature of local shape properties was used by the authors.

B. DEEP LEARNING-BASED APPROACHES

Lately, deep learning methods have been broadly investigated in 3D model retrieval and classification tasks. They are applied on various 3D object representations, especially points, voxels and 2D projections.

PointNet [26] is deemed the pioneer in utilizing the point cloud, as an input data where each of its points is characterized using the (x, y, z) coordinates. In the pre-processing phase, feature transformation and inputs are fed into the PointNet framework. PointNet consists of three major modules: a "Spatial Transformer Network (STN)" module, a RNN module and a simple symmetric function that combines all the information from each point in the point cloud. For an accurate capturing of local structures, the same authors posit a hierarchal PointNet which is named PointNet++ [27]. Ng *et al.* [28] present RadialNet, a novel deep neural network framework, which takes full advantage of local structure representation of point cloud data by applying radial basis function.

Wu *et al.* [29] introduced a pioneering method of using volumetric CNN for 3D model classification. To characterize a 3D geometric form as a probability distribution of binary variables on a 3D voxel grid, the suggested 3D ShapeNets makes use of a convolutional deep neural network. A similar method was presented by Maturana and Scherer [30], called VoxNet. The latter probed three different occupancy grid models with 3D CNN for real-time and effective object recognition. It significantly outperformed 3D ShapeNets on the ModelNet database. Qi *et al.* [31] developed two novel volumetric 3D convolutional neural networks by joining anisotropic probing, auxiliary training, and network in network structure.

Su *et al.* [32] (MVCNN) characterize a 3D object by views produced from the projection at twelve varying standpoints, and then utilize VGG-M convolution neural network to learn the characteristics of each view, finally, the characteristics of the multi-view are joined and sent to the next CNN network to generate the final shape descriptors. MVCNN considerably outperforms any results that were previously published. Johns *et al.* [33] implement CNN to generic multi-view recognition by partitioning an image sequence into an ensemble of image pairs, categorizing each pair, and weighing its contribution. DeepPano [34] was introduced to learn characteristics from panorama views making use of CNN, where each panorama view can be considered as the unified combination of many views captured on a circle. Row-wise max pooling was proposed in DeepPano to eradicate the impact of rotation about the up-orientation. With pose normalization, Sfikas *et al.* [35] employed CNN to learn 3D global characteristics from many panorama views which were piled together in a consistent order.

III. CONVOLUTIONAL NEURAL NETWORKS

To begin with, we review a few key features of CNNs which were presented by LeCun *et al.* [36]. It is known that 2D CNNs are commonly used in speech and image domains. As such, we expand 2D convolution to 3D with depth axis and utilize 3D convolution to build the engineering of our 2DSlicesNet.

A. BACKGROUND OF 2D CONVOLUTIONAL NEURAL NETWORKS

To pull out features from the previous layer in 2D convolutional neural networks, 2D convolution is effectuated at the convolutional layers. The concept is that a filter, named local receptive field, shifts over every unit from previous layer. Every unit in the convolutional layer gets inputs from an ensemble of units situated in the local receptive field and is computed by the following equation.

$$f_{xy} = \sigma \left(\sum_{i,j} w_{ij} v_{(x+i)(y+j)} + b \right) \quad (1)$$

where f_{xy} refers to a unit in feature map at location $(x; y)$, $\sigma(\bullet)$ symbolizes an activation function, w_{ij} signifies the weight of filter, $v_{(x+i)(y+j)}$ designates an input unit at location $(x + i, y + j)$, and b refers to the bias of the feature map. Parameters of filters are required to be the same for all of the possible positions of precedent layer. This process is named weight sharing. The weight sharing diminishes the quantity of free variables and rises the generalization ability of the network. The weights of filters are reproduced over the input data, generating intrinsic insensibility to translation in the input. In order to detect various features, the convolutional layer commonly regroups many feature maps. To improve the invariance to distortions on the inputs, in the subsampling layers, the feature maps resolution is decreased by pooling over local neighbourhood on the feature maps in the preceding layer. A CNN structure can be built

by assembling, in an alternating manner, several convolution and subsampling layers. The habitual back propagation gradient-descent technique is used to train the networks.

B. 3D CONVOLUTION

In 2D CNNs, the features are extracted from the two dimensions only (length and height) by applying convolutions on the 2D feature maps. When applied to 3D object recognition, it is commonly aimed at extracting the features encoded in three dimensions of the input data (length, height, and depth). Consequently, we propose to perform 3D convolutions in the convolution steps to capture information from the three dimensions. In our approach, the 3D convolution is realized by convolving a 3D filter to the cube made by stacking a set of successive 2D slices together. By this structure, in the convolution layer, the feature maps are associated with several contiguous 2D slices in the preceding layer. Hence, we extract features from both length, height, and depth. Like the equation 1, 3D convolution is computed by:

$$f_{xyz} = \sigma \left(\sum_{i,j,k} w_{ijk} v_{(x+i)(y+j)(z+k)} + b \right) \quad (2)$$

Because 3D convolutional filter extracts only one kind of features from the frame cube, the weights of filter are reproduced across the whole cube. Generally, in late layers, the number of feature maps should be increased by producing multiple sorts of features based on the same set of previous feature maps. Like the 2D convolution, we can apply several 3D convolutions with different filters to the same position in the preceding layer.

IV. OUR APPROACH

Based on the 3D convolution mentioned above, a variety of CNN structures can be developed. Hence, we present a 3D CNN architecture based on 2D slices that we have devised for 3D objects classification and retrieval. In our approach, presented in Fig.1, we start by a normalization step, to assure that similar models will be in the same orientation, position and scale. Next, the 2D slices corresponding to the first main axis of each 3D model are extracted, resized, stacked and then employed as input to our 3DCNN.

In order to compare 3D objects, we have to get their characteristics. The last fully connected layer of our network is used to obtain the 3D objects' features. Finally, the Euclidean distance is used to calculate the similarity between 3D models. The details will be presented in the following subsections.

A. 3D OBJECT PRETREATMENT

3D models obtained by diverse scanning and modeling frameworks have various coordinate frames, because the shapes of 3D models are always formed in a specific coordinate system. For several applications, such as content-based retrieval, visualization, thumbnail generation and modelling, the coordinate frame normalization usually needs to be completed at first. For example, numerous 3D object

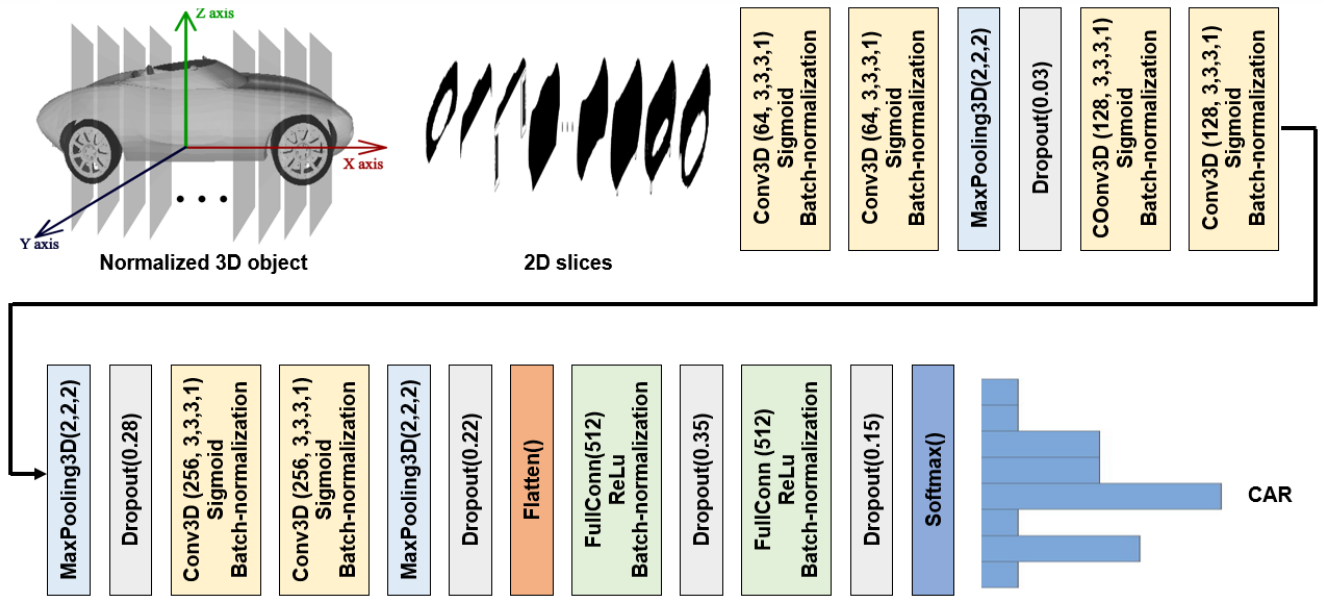


FIGURE 1. The overview of 2DSlicesNet. The 2D slices corresponding to the first main axis of the normalized 3D object are extracted, resized, stacked and then used as the input data of our 3D CNN.

classification and retrieval approaches necessitate a prior normalization step for a given request object and the 3D objects of database, so all of the normalized 3D objects are aligned into a usual coordinate framework before they are treated.

In fact, the normalization step includes the normalization of the scale, translation and rotation. In our approach, the normalization step consists only of scaling and translating each 3D model, in view of the fact that 3D objects of both ModelNet10 and ModelNet40 are manually aligned by the Princeton’s team. To accomplish the scale normalization, the average distance of a 3D object’s surface from its centroid is equal to 1. The translation normalization is achieved by computing the 3D object’s center of mass and translates it to coincide with the origin.

B. 2D SLICES EXTRACTION

In order to get the 3D objects’ slices, we extract the intersection of the 3D triangle mesh with 64 planes equally spaced and orthogonal to its first main axis. In fact, we move the radius in the associated plane and we calculate, each time, the distance D between the 3D triangle mesh and the intersection with the radius’s origin O .

Let us consider I the intersection point of the radius oriented by the vector \vec{v} and a triangle mesh ABC . The following equation defines the point I :

$$OI = D \cdot \vec{v} \tag{3}$$

The following equation checked that the intersection point I is in the surface delimited by the facet ABC :

$$\vec{OA} \cdot \vec{n} = \vec{OI} \cdot \vec{n} \tag{4}$$

With \vec{n} referring to the normal vector to the triangle ABC , the following relation determines it:

$$\vec{n} = \frac{\vec{AB} \wedge \vec{AC}}{\|\vec{AB} \wedge \vec{AC}\|} \tag{5}$$

To assure that the point of intersection I is not empty, it is enough that it checks the following conditions:

$$\begin{cases} (\vec{IA} \wedge \vec{IB}) \cdot \vec{n} > 0 \\ (\vec{IB} \wedge \vec{IC}) \cdot \vec{n} > 0 \\ (\vec{IC} \wedge \vec{IA}) \cdot \vec{n} > 0 \end{cases} \tag{6}$$

All the extracted 2D slices are first resized into 64×64 and then stacked into a matrix. At the end of this process, each 3D object is represented by an ensemble of 64 slices, which we will use as the input data of our 3D CNN. An example of a 3D object is shown in Fig.2 with its 2D slices corresponding to its x-axis using 2DSlicesNet.

C. DATA AUGMENTATION

Data augmentation includes a set of practices that allow us to meaningfully increase the variety of data existing for training approaches, without actually assembling new data. As a matter of fact, data augmentation techniques enable us to increase the size and quality of training datasets and to provide some variety as better deep learning approaches can be constructed using them, which rise generalization performance and reduce over-fitting.

Data rotation is one of the most commonly used techniques of data augmentation in CNN. Thus, we made use of this technique in order to better train our network. Actually, 3D objects in the used databases are upright oriented. However, they are not regularly oriented along the axis; they could

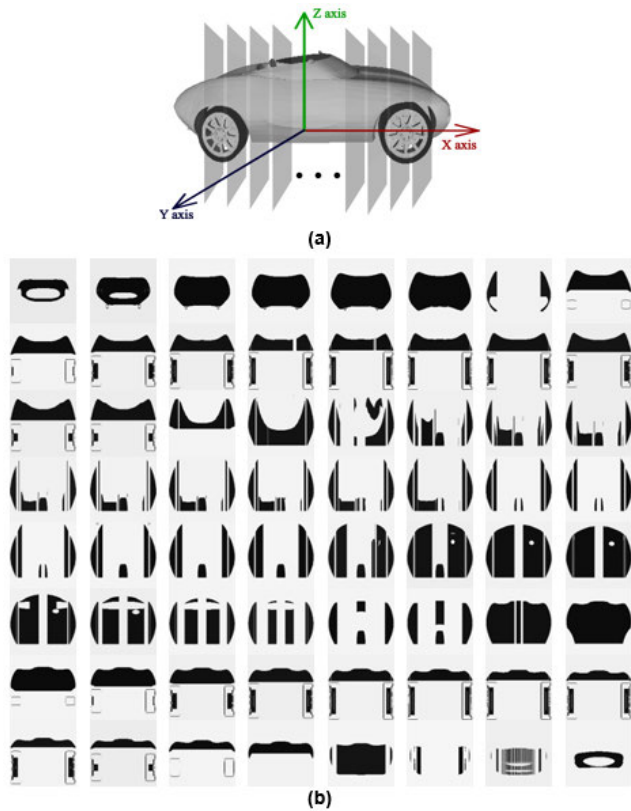


FIGURE 2. Example of a 3D object (a) with its 2D slices (b) corresponding to its x-axis using 2DSlicesNet.

be rotated randomly along the upright direction. Since our approach relies on 2D slices of 3D objects, the input data can profit from rotation augmentation on the upright axis for training, and optionally, for testing. During training time, we rotate the 2D slices of each 3D object by 11.25 degree, and we consider each rotated version as different training instance. At testing time, the 3D objects' slices and all their rotated versions are fed into our 3DCNN in one batch, and the feature maps of the last fully connected layer are averaged.

D. 2DSlicesNet ARCHITECTURE

The 2DSlicesNet architecture is based on an ordinary scheme, specifically an input layer, six convolution layers, three pooling layers, two fully connected layers, and a Softmax output layer.

For all convolutional layers, we use a convolutional filter of size $3 \times 3 \times 3$ followed by Sigmoid non-linearity function, the correspondent feature maps are 64, 64, 128, 128, 256 and 256, respectively. To accelerate network training, Batch-normalization [37] is employed after each convolutional layer.

After each two convolutional layers, we integrate $2 \times 2 \times 2$ max-pooling layer to make the feature maps unsusceptible to small translations and to reduce the resolution of the feature maps. In order to improve the generalization capability and avoid over-fitting, we add dropout after each pooling layer.

The choice of using small receptive fields ($3 \times 3 \times 3$), using stride 1, instead of employing comparatively wide receptive fields in the first convolutional layers, is supported by the idea that a stack of two $3 \times 3 \times 3$ convolutional layers, without pooling layer in between, has a real receptive field of $5 \times 5 \times 5$. In fact, when we use, for example, a stack of two $3 \times 3 \times 3$ convolutional layers, rather than a single $5 \times 5 \times 5$ convolutional layer; firstly we reduce the parameters' number, secondly we integrate two nonlinear rectification layers rather than only one, increasing the decision function's discriminability.

The output of our architecture includes two fully connected layers, each has 512 output units. The two fully connected layers are followed by ReLu function and dropout layer. Finally, a Softmax layer is used at the end of our system to allow for the parameter optimization by minimizing the classification errors of 3D models. The class with the highest probability is treated like the predicted class for the 3D model. The details of our architecture are shown in Fig.1.

E. FEATURE EXTRACTION FOR RETRIEVAL TASK

The descriptor for the retrieval task is the output of the last fully connected layer after ReLU activation function of our 2DSlicesNet, which is a 512 dimensional vector. Each 3D object descriptor is compared against the rest of the 3D object descriptors using the Euclidean distance. For two 3D objects Q and O , their descriptors are extracted from 2DSlicesNet V_q and V_o , respectively. The Euclidean distance metric formula is determined as:

$$d(Q, O) = \|V_q - V_o\|_2 \quad (7)$$

V. EXPERIMENTS

In this section, we first introduce the datasets used to evaluate our method. Then, a brief account of the implementation details is provided. Next, we present the experiments on 3D object classification and retrieval. Also, we discuss the results of the experiments and compare them with well-known approaches. It is worth mentioning that the scores of the compared approaches are those declared by the authors in the respective papers.

A. IMPLEMENTATION DETAILS

The 2DSlicesNet was evaluated on an Intel Xeon E5-2670 CPU system, with 64 Go of RAM and a NVIDIA QUADRO M6000 GPU with 12 GB RAM. The 2D slices extraction approach was implemented in C++, while the network architecture was developed in Python 3.7.7 using Keras 2.3.1 and Tensorflow 2.1.0 via CUDA instruction set on the GPU.

For hyperparameter optimization, we used Hyperas, which chooses the best performing parameters out of the options given. As far as the algorithm parameter is concerned, Hyperas opted for Stochastic Gradient Descent with 0.9 momentum [38], instead of Adam and RMSProp. For the learning rate, it selected 0.001 out of 0.01 and 0.1. For

the activation function, Hyperas chose Sigmoid function for the six convolutional layers. However, it opted for the ReLu function for the two fully connected layers. Also, we made use of Hyperas to select the best value for the dropout layers within [0, 0.9]. Finally, the batch size was 32. It is worth noting that we trained our network for 200 epochs.

B. DATASETS

In our experiments, we tested our approach on the two versions of the Princeton ModelNet dataset [29], ModelNet10 and ModelNet40, which are two commonly used subsets in ModelNet. The ModelNet10 contains 4,899 3D objects from 10 classes, while the ModelNet40 is comprised of 12,311 3D objects from 40 classes. All of the 3D objects in both ModelNet10 and ModelNet40 are cleaned and manually aligned by the Princeton’s team. In our tests, the training and testing databases are divided following the same setting, outlined in [29].

C. 3D OBJECT CLASSIFICATION

We have first evaluated our 2DSlicesNet method in classification task on the ModelNet10 and ModelNet40’s test subset. We have compared our approach with recent state-of-the-art approaches, including methods that do not use machine learning; Spherical Harmonics (SPH) [23] and Light Field (LFD) [18] descriptors, and those that use machine learning, namely RadialNet [28], LonchaNet [8], Primitive-GAN [39], VSL [40], BinVoxNetPlus [41], DeepSets [42], 3D-DescriptorNet [43], the approach proposed by Soltani et al. [44], the approach presented by Zanuttigh and Minto [45], ECC [46], FPNN [47], PointNet [26], PointNet [48], LightNet [49], the approach proposed by Xu and Todorovic [50], Geometry Image [51], 3D-GAN [52], Pairwise [33], MVCNN [32], GIFT [53], VoxNet [30], DeepPano [34] and 3DShapeNets [29]. Table.1 recapitulates the classification accuracy of the abovementioned approaches.

Our 2DSlicesNet approach outperforms all compared approaches in the ModelNet40 database. However, in ModelNet10 database, it is only outperformed by the LonchaNet approach [8] by a small margin (94.05% vs 94.37%). The LonchaNet is only based on three 2D slices corresponding to the 3D object’s main axes, which can cause it to represent some 3D models of the same category by 2D slices that are completely dissimilar, or the opposite, similar 2D slices for 3D models of different classes. The LonchaNet’s authors did not test their method on a large database with more categories, because we think that, if we increase the number of objects and classes, we risk falling into the problem mentioned above.

Fig.3 shows the confusion matrix for ModelNet10’s test split. As we can observe, the approach precisely classifies the 3D objects except a few, which seem similar from a visual perspective. As we can see in Fig.3, the majority of the misclassification originates from the similar class pairs such as dresser versus night stand and desk versus table.

Fig.4 demonstrates the classification precision-recall plots for the different classes of the ModelNet10’s test split.

TABLE 1. Classification accuracy results reached by our approach (2DSlicesNet) and some state-of-the-art approaches on the Modelnet10 and Modelnet40. Approaches that do not use machine learning are indicated by (NON-ML). ‘-’ means that no information is presented for the corresponding method in the corresponding paper.

Approaches	ModelNet10	ModelNet40
2DSlicesNet(Ours)	94.05%	91.05%
LonchaNet [8]	94.37%	-
RadialNet [28]	89.53%	86.42%
Primitive-GAN [39]	92.20%	86.40%
VSL [40]	91.00%	84.50%
BinVoxNetPlus [41]	92.32%	85.47%
DeepSets [42]	-	90.30%
3D-DescriptorNet [43]	92.40%	-
Soltani et al. [44]	-	82.10%
Zanuttigh and Minto [45]	91.50%	87.80%
ECC [46]	90.00%	83.20%
FPNN [47]	-	88.40%
PointNet [26]	-	89.20%
PointNet [48]	77.60%	-
LightNet [49]	93.94%	88.93%
Xu and Todorovic [50]	88.00%	81.26%
Geometry Image [51]	88.40%	83.90%
3D-GAN [52]	91.00%	83.30%
Pairwise [33]	92.80%	90.70%
MVCNN [32]	-	90.10%
GIFT [53]	92.35%	83.10%
VoxNet [30]	92.00%	83.00%
DeepPano [34]	85.45%	77.63%
3DShapeNets [29]	83.50%	77.00%
LFD (NON-ML) [18]	79.87%	75.47%
SPH (NON-ML) [23]	79.79%	68.23%

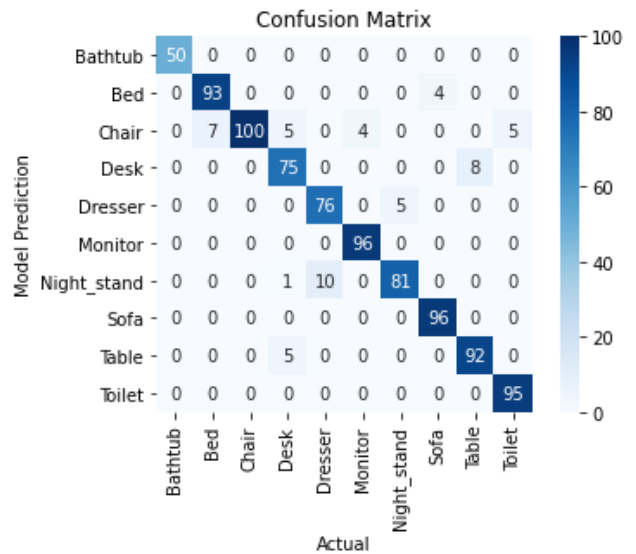


FIGURE 3. Confusion matrix of the classification results attained by our 2DSlicesNet using ModelNet10 database.

The shown precision-recall curves exhibit the strangeness and stability of our approach in terms of 3D objects’ classification task. In fact, all the classes achieve 1.00 in average precision (AP) except Desk (0.96), Table (0.97), Dresser (0.98) and Night Stand (0.97), confirming the outcomes found in Fig.3.

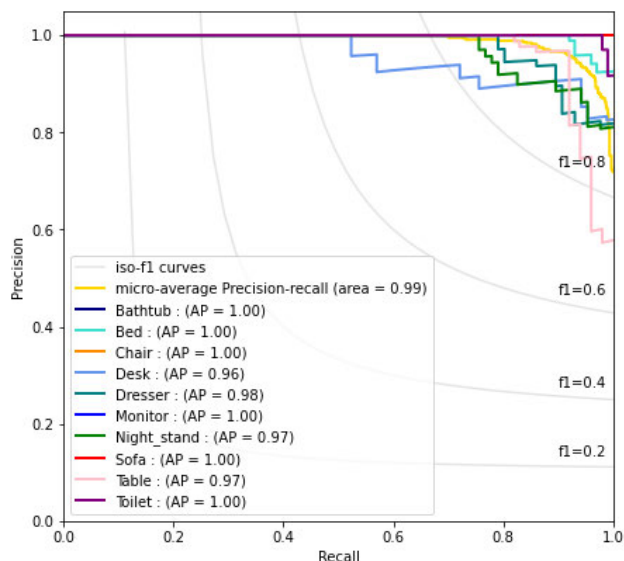


FIGURE 4. Precision-recall curves of classification for each class of the ModelNet10 database. 'AP' means the micro average precision corresponding to each ModelNet10's class.

D. 3D OBJECT RETRIEVAL

In order to examine the performance of our approach on the task of 3D object retrieval, we carried out additional evaluation. The effectiveness of our 2DSlicesNet system to retrieve 3D object was measured on the ModelNet10 and ModelNet40 databases, by using the Precision-Recall curves and the mean Average Precision (mAP) score, against some well-known approaches that provide retrieval results. More precisely, Spherical Harmonics (SPH) [23], Light Field (LFD) [18], Geometry Image [51], PANORAMA [21] and 3DShapeNets [29]. The findings of the aforementioned approaches are those stated by the researchers in their own papers. Table.2 outlines the retrieval experiment's results in terms of mAP where our 2DSlicesNet outperforms the compared approaches in both databases. As we can notice, the proposed approach ranks number one followed by Geometry Image [51].

TABLE 2. Comparison of retrieval results on the ModelNet10 and ModelNet40 databases measured in mean Average Precision (mAP). Approaches that do not use machine learning are indicated by (NON-ML).

Approaches	ModelNet10	ModelNet40
2DSlicesNet (Ours)	76.36%	72.08%
Geometry Image [51]	74.9%	51.3%
3D ShapeNets [29]	68.3%	49.2%
PANORAMA (NON-ML) [21]	60.32%	46.13%
LFD (NON-ML) [18]	49.82%	40.91%
SPH (NON-ML) [23]	44.05%	33.26%

Fig.5 and Fig.6 plot recall-precision curves for our approach, 3D ShapeNets [29], PANORAMA [21], LFD [18] and SPH [23] on ModelNet10 and ModelNet40, respectively.

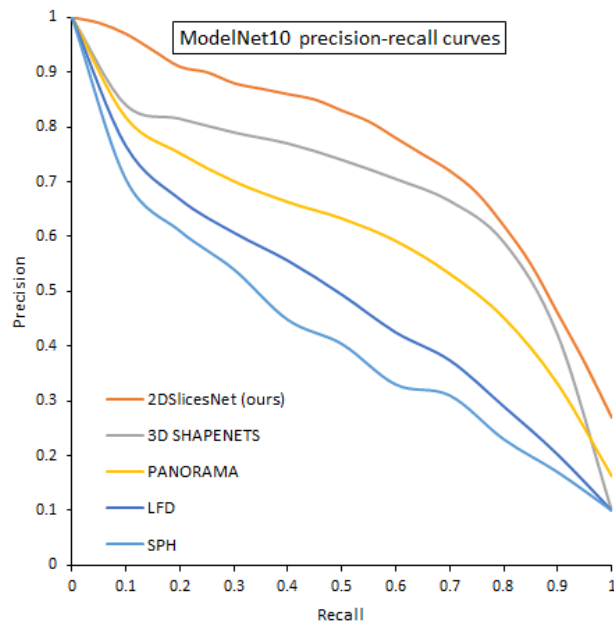


FIGURE 5. Precision-recall curves for ModelNet10 subsets. Shown are our proposed approach (2DSlicesNet) compared to four well-known retrieval approaches.

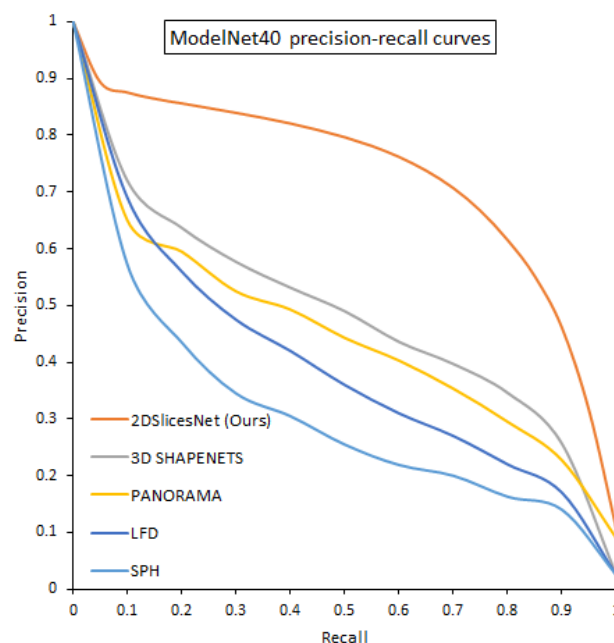


FIGURE 6. Precision-recall curves for ModelNet40 subsets. Shown are our proposed approach (2DSlicesNet) compared to four well-known retrieval approaches.

As we can observe, the recall-precision curves exhibit the efficacy of our proposed method in 3D objects retrieval task by surpassing all the compared methods, and confirm the mAP scores presented in Table.2. Moreover, compared to the other approaches, our method's whole curve decreases slowly, when the recall increases in the two database, which proves that the approach is more stable. Particularly on the ModelNet40, the suggested approach keeps roughly the same

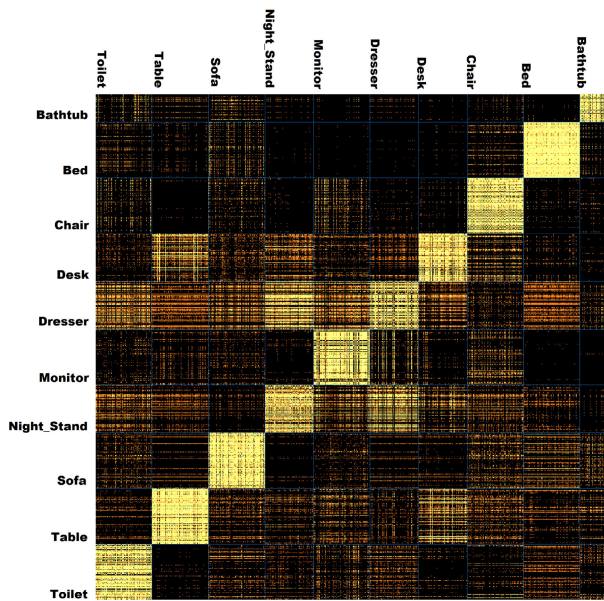


FIGURE 7. Tier image visualizing nearest neighbor (white), first tier (yellow), and second tier (orange) computed by matching every 3D object (rows) with every other 3D object (columns) in the ModelNet10 database using 2DSlicesNet.

curve, while the curve of the compared approaches regress significantly, which means that our approach performs well even if we increase the number of categories.

Fig. 7 shows the tier image for the 3D objects of the ModelNet10 database. White, yellow and orange visualize nearest neighbor, first tier, and second tier matches, respectively. A powerful retrieval method should have a set of white-yellow pixels in the class-sized blocks along the diagonal. It is noticeable that our approach has brighter pixels, especially yellow pixels, in the diagonal class-sized blocks. This demonstrates that the 3D objects belong to the same category indicate higher similarity. We also notice other orange blocks without the diagonals class-sized blocks especially the desk and table classes. In fact, desk and table categories generally have similar structure and particularly some desks have minor visual characteristics that differentiate them from table, which makes the system confuse them.

VI. CONCLUSION

In this paper, we propose 2DSlicesNet, a 2D slice-based CNN approach for 3D object retrieval and classification tasks. We use 2D slices of 3D objects as input data to a 3D CNN, which aggregates the information within 2D slices in a robust and discriminative descriptor. The performance of the 2DSlicesNet is reinforced with data augmentation and network architecture. The usefulness and the effectiveness of 2DSlicesNet was demonstrated on the ModelNet10 and ModelNet40 datasets, achieving competitive performance in a set of experiments. In our ongoing research, we will

continue to investigate 2D slice-based approaches by developing other deep learning architectures.

REFERENCES

- [1] Z. Gao, S. H. Li, G. T. Zhang, Y. J. Zhu, C. Wang, and H. Zhang, "Evaluation of regularized multi-task learning algorithms for single/multi-view human action recognition," *Multimedia Tools Appl.*, vol. 76, no. 19, pp. 20125–20148, Oct. 2017.
- [2] A.-A. Liu, W.-Z. Nie, Y. Gao, and Y.-T. Su, "Multi-modal clique-graph matching for view-based 3D model retrieval," *IEEE Trans. Image Process.*, vol. 25, no. 5, pp. 2103–2116, May 2016.
- [3] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Trans. Graph.*, vol. 21, no. 4, pp. 807–832, 2002.
- [4] A. Mademlis, P. Daras, D. Tzovaras, and M. G. Strintzis, "3D object retrieval using the 3D shape impact descriptor," *Pattern Recognit.*, vol. 42, no. 11, pp. 2447–2459, Nov. 2009.
- [5] W. Mohamed and A. B. Hamza, "Deformable 3D shape retrieval using a spectral geometric descriptor," *Appl. Intell.*, vol. 45, no. 2, pp. 213–229, Sep. 2016.
- [6] B. Leng, Z. Qin, X. Cao, T. Wei, and Z. Zhang, "Mate: A visual based 3D shape descriptor," *Chin. J. Electron.*, vol. 18, no. 2, pp. 291–296, 2009.
- [7] H. Ghodrati and A. B. Hamza, "Nonrigid 3D shape retrieval using deep auto-encoders," *Appl. Intell.*, vol. 47, no. 1, pp. 44–61, 2017.
- [8] F. Gomez-Donoso, A. Garcia-Garcia, J. Garcia-Rodriguez, S. Orts-Escolano, and M. Cazorla, "LonchaNet: A sliced-based CNN architecture for real-time 3D object recognition," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 412–418.
- [9] X. Zhao, H. Zhang, Y. Jiang, S. Song, X. Jiao, and M. Gu, "An effective heuristic-based approach for partitioning," *J. Appl. Math.*, vol. 2013, pp. 1–8, Apr. 2013.
- [10] R. Hong, Z. Hu, R. Wang, M. Wang, and D. Tao, "Multi-view object retrieval via multi-scale topic models," *IEEE Trans. Image Process.*, vol. 25, no. 12, pp. 5814–5827, Dec. 2016.
- [11] Y. Gao and Q. Dai, "View-based 3D object retrieval: Challenges and approaches," *IEEE Multimedia Mag.*, vol. 21, no. 3, pp. 52–57, Jul. 2014.
- [12] D. Wang, B. Wang, S. Zhao, H. Yao, and H. Liu, "View-based 3D object retrieval with discriminative views," *Neurocomputing*, vol. 252, pp. 58–66, Aug. 2017.
- [13] I. O. Taybi, R. Alaoui, F. R. Zakani, K. Arhid, M. Bouksim, and T. Gadi, "A novel efficient 3D object retrieval method based on representative slices," in *Proc. 5th Int. Conf. Multimedia Comput. Syst. (ICMCS)*, Sep. 2016, pp. 639–644.
- [14] I. O. Taybi, M. Bouksim, R. Alaoui, and T. Gadi, "A novel partial 3D object retrieval method using adaptive slices clustering," *Int. J. Intell. Eng. Syst.*, vol. 12, no. 1, pp. 253–262, Feb. 2019.
- [15] S. E. Naffouti, Y. Fougerolle, I. Aouissau, A. Sakly, and F. Mériaudeau, "Heuristic optimization-based wave kernel descriptor for deformable 3D shape matching and retrieval," *Signal, Image Video Process.*, vol. 12, no. 5, pp. 915–923, Jul. 2018.
- [16] D. Saupé and D. V. Vranić, "3D model retrieval with spherical harmonics and moments," in *Proc. Joint Pattern Recognit. Symp.* Berlin, Germany: Springer, 2001, pp. 392–397.
- [17] L. Nie, M. Wang, Z.-J. Zha, and T.-S. Chua, "Oracle in image search: A content-based approach to performance prediction," *ACM Trans. Inf. Syst.*, vol. 30, no. 2, pp. 1–23, May 2012.
- [18] D.-Y. Chen, X.-P. Tian, Y.-T. Shen, and M. Ouhyoung, "On visual similarity based 3D model retrieval," *Comput. Graph. Forum*, vol. 22, no. 3, pp. 223–232, Sep. 2003.
- [19] Y. Gao, J. Tang, R. Hong, S. Yan, Q. Dai, N. Zhang, and T.-S. Chua, "Camera constraint-free view-based 3-D object retrieval," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 2269–2281, Apr. 2012.
- [20] I. O. Taybi, T. Gadi, and R. Alaoui, "A new partial 3D object indexing and retrieval approach combining 2D slices and apriori algorithm," *Sci. Visualizat.*, vol. 12, no. 2, pp. 110–126, 2020.
- [21] P. Papadakis, I. Pratikakis, T. Theoharis, and S. Perantonis, "PANORAMA: A 3D shape descriptor based on panoramic views for unsupervised 3D object retrieval," *Int. J. Comput. Vis.*, vol. 89, nos. 2–3, pp. 177–192, Sep. 2010.
- [22] E. A. Lmaati, A. El Oirak, and M. N. Kaddiou, "A 3D search engine based on 3D curve analysis," *Signal, Image Video Process.*, vol. 4, no. 1, pp. 89–98, Mar. 2010.

- [23] 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.
- [24] A. Tatsuma and M. Aono, "Multi-Fourier spectra descriptor and augmentation with spectral clustering for 3D shape retrieval," *Vis. Comput.*, vol. 25, no. 8, pp. 785–804, Aug. 2009.
- [25] C. B. Akgül, B. Sankur, Y. Yemez, and F. Schmitt, "3D model retrieval using probability density-based shape descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 1117–1133, Jun. 2009.
- [26] 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 Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 652–660.
- [27] 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. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.
- [28] Y. T. Ng, C. M. Huang, Q. T. Li, and J. Tian, "RadialNet: A point cloud classification approach using local structure representation with radial basis function," *Signal, Image Video Process.*, vol. 14, no. 4, pp. 747–752, Jun. 2020.
- [29] 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. (CVPR)*, Jun. 2015, pp. 1912–1920.
- [30] D. Maturana and S. Scherer, "VoxNet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 922–928.
- [31] C. R. Qi, H. Su, M. NieBner, 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. (CVPR)*, Jun. 2016, pp. 5648–5656.
- [32] 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. (ICCV)*, Dec. 2015, pp. 945–953.
- [33] 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. (CVPR)*, Jun. 2016, pp. 3813–3822.
- [34] 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.
- [35] K. Sfikas, T. Theoharis, and I. Pratikakis, "Exploiting the panorama representation for convolutional neural network classification and retrieval," *3DOR*, vol. 6, p. 7, Apr. 2017.
- [36] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [37] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
- [38] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.
- [39] S. H. Khan, Y. Guo, M. Hayat, and N. Barnes, "Unsupervised primitive discovery for improved 3D generative modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9739–9748.
- [40] S. Liu, L. Giles, and A. Ororbia, "Learning a hierarchical latent-variable model of 3D shapes," in *Proc. Int. Conf. 3D Vis. (3DV)*, Sep. 2018, pp. 542–551.
- [41] C. Ma, Y. Guo, Y. Lei, and W. An, "Binary volumetric convolutional neural networks for 3-D object recognition," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 1, pp. 38–48, Jan. 2019.
- [42] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Póczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3391–3401.
- [43] J. Xie, Z. Zheng, R. Gao, W. Wang, S.-C. Zhu, and Y. N. Wu, "Learning descriptor networks for 3D shape synthesis and analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8629–8638.
- [44] A. A. Soltani, H. Huang, J. Wu, T. D. Kulkarni, and J. B. Tenenbaum, "Synthesizing 3D shapes via modeling multi-view depth maps and silhouettes with deep generative networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1511–1519.
- [45] P. Zanuttigh and L. Minto, "Deep learning for 3D shape classification from multiple depth maps," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 3615–3619.
- [46] M. Simonovsky and N. Komodakis, "Dynamic edge-conditioned filters in convolutional neural networks on graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3693–3702.
- [47] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas, "FPNN: Field probing neural networks for 3D data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 307–315.
- [48] A. Garcia-Garcia, S. Orts-Escolano, F. Gomez-Donoso, M. Cazorla, J. Garcia-Rodriguez, and J. Azorin-Lopez, "PointNet: A 3D convolutional neural network for real-time object class recognition," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 1578–1584.
- [49] S. Zhi, Y. Liu, X. Li, and Y. Guo, "Toward real-time 3D object recognition: A lightweight volumetric CNN framework using multitask learning," *Comput. Graph.*, vol. 71, pp. 199–207, Apr. 2018.
- [50] X. Xu and S. Todorovic, "Beam search for learning a deep convolutional neural network of 3D shapes," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 3506–3511.
- [51] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3D shape surfaces using geometry images," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 223–240.
- [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. Adv. Neural Inf. Process. Syst.*, 2016, pp. 82–90.
- [53] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. J. Latecki, "GIFT: A real-time and scalable 3D shape search engine," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5023–5032.



ILYASS OUAZZANI TAYBI received the master's degree in computer science and distributed systems from the Faculty of Science, University Ibn Zohr, Agadir, Morocco, in 2013. He is currently pursuing the Ph.D. degree with the Informatics, Imaging, and Modeling of Complex Systems Laboratory, Settat, Morocco. His research interests include 3D models classification and retrieval, datamining and database analysis, and machine learning.



TAOUFIQ GADI is currently a Professor of computer science with the Faculty of Science and Technologies, Hassan First University of Settat, Morocco. Since 2014, he has been the Director of the Informatics, Imaging, and Modeling of Complex Systems Laboratory. He has conducted more than twenty Ph.D. theses and written a seventy of scientific articles in the domain of 3D models analysis, models Driving Architecture, Datamining and Database Analysis, Modeling of Complex Systems, and Machine Learning.



RACHID ALAOUI was born in Morocco, in 1977. He is currently an Associate Professor and a member of the Research Laboratory for Computer Science and Telecom, Mohamed V University, Rabat, and holds professional membership in well recognized international committees and reviewing committees. His research interests include the data science, machine learning, 3D models recognition, and technological innovation.