Multi-View 3D Object Retrieval With Deep Embedding Network

Haiyun Guo, Jinqiao Wang, *Member, IEEE*, Yue Gao, *Senior Member, IEEE*, Jianqiang Li, and Hanqing Lu, *Senior Member, IEEE*

***Abstract*— In multi-view 3D object retrieval, each object is characterized by a group of 2D images captured from different views. Rather than using hand-crafted features, in this paper, we take advantage of the strong discriminative power of convolutional neural network to learn an effective 3D object representation tailored for this retrieval task. Specifically, we propose a deep embedding network jointly supervised by classification loss and triplet loss to map the high-dimensional image space into a low-dimensional feature space, where the Euclidean distance of features directly corresponds to the semantic similarity of images. By effectively reducing the intra-class variations while increasing the inter-class ones of the input images, the network guarantees that similar images are closer than dissimilar ones in the learned feature space. Besides, we investigate the effectiveness of deep features extracted from different layers of the embedding network extensively and find that an efficient 3D object representation should be a tradeoff between global semantic information and discriminative local characteristics. Then, with the set of deep features extracted from different views, we can generate a comprehensive description for each 3D object and formulate the multi-view 3D object retrieval as a set-to-set matching problem. Extensive experiments on SHREC'15 data set demonstrate the superiority of our proposed method over the previous state-of-the-art approaches with over 12% performance improvement.**

***Index Terms*— Convolutional neural network, multi-view 3D object retrieval, triplet loss.**

I. Introduction

DUE to the rapid developments of 3D acquisition techniques combined with the popular applications of computer graphics and vision, recent years have witnessed an exponential increase in the number of 3D object models across various fields [1]–[3]. The explosive growth of 3D object collections has made it crucial to develop effective algorithms for content-based 3D object retrieval. Generally, there are two paradigms for 3D object retrieval [4]: 3D model-based methods and 2D view-based ones. 3D model-based methods [5], [6] generate object descriptors directly from the information of 3D models, e.g., geometric moment, surface distribution, and voxel-based features. However, the difficulty in obtaining 3D models of real objects and the expensive computation of 3D model reconstruction have severely limited the application of model-based methods. In contrast, 2D view-based methods [7], [8] represent each 3D object with a group of 2D images captured from different views and build object descriptors by extracting various visual features from these images. With the great flexibility of acquiring 2D images for real 3D objects from multiple views, view-based methods have attracted much research attention.

The challenges of multi-view 3D object retrieval lie in view selection, visual feature extraction and similarity measurement [9]. To begin with, rather than selecting representative views for each 3D object [7], [10], [11], in this paper, we target on exploiting all the views to learn a comprehensive 3D object description. Furthermore, abstracting discriminative visual features from 2D images is essential for multi-view 3D object retrieval. It is also a very challenging step due to the following factors. On the one hand, the intra-class variations of 2D images are quite large. Because the appearances of different 3D objects within the same class can be rather different and the same 3D object can have diverse 2D images due to illumination changes and view-point shifts. On the other hand, the inter-class variations may be not large enough since two 3D objects from different classes might look similar in some views. Sometimes the intra-class variations can even overwhelm inter-class ones. Therefore, it is crucial to develop effective visual features to reduce intra-class variations while enlarge inter-class ones. After representing each 3D object with a set of visual features extracted from multiple views, the original retrieval task is converted to a set-to-set matching problem, the key part of which is how to measure the distance between two sets.

Most of existing view-based methods, e.g., Lighting Field Descriptor (LFD) [12], PANORAMA [13], and compact multi-view descriptors [7], focus on the primordial characteristics of 3D objects, such as Zernike moments [14] and Fourier

Manuscript received April 5, 2016; revised August 28, 2016; accepted September 10, 2016. Date of publication September 15, 2016; date of current version October 3, 2016. This work was supported in part by the 863 Program under Grant 2014AA015104, and in part by the National Natural Science Foundation of China under Grant 61273034 and Grant 61332016. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Aydin Alatan. *(Corresponding author: Jinqiao Wang.)*

H. Guo, J. Wang, and H. Lu are with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, and also with the University of Chinese Academy of Sciences, Beijing 100190, China (e-mail: haiyun.guo@nlpr.ia.ac.cn; jqwang@nlpr.ia.ac.cn; luhq@nlpr.ia.ac.cn).

Y. Gao is with the Key Laboratory for Information System Security, Ministry of Education, Tsinghua National Laboratory for Information Science and Technology (TNList), School of Software, Tsinghua University, Beijing 100084, China (e-mail: kevin.gaoy@gmail.com).

J. Li is with the Beijing Engineering Research Center for IoT Software and Systems, School of Software Engineering, Beijing University of Technology, Beijing 100083, China (e-mail: lijianqiang@bjut.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2016.2609814

descriptors [15], or common local visual features, such as SIFT [16] and HoG [17]. Since deep learning methods have demonstrated to be more effective than hand-crafted descriptors in abstracting discriminative features and have achieved great success in many visual recognition tasks [18]–[20], some researchers tended to take advantage of deep neural networks for view-based 3D object retrieval. For example, Zhu *et al.* [8] utilized auto-encoder to learn 3D shape representations from a set of depth images with multiple views. Considering the great power of Convolutional Neural Network (CNN) in sematic feature abstraction, we leveraged CNN pre-trained for image classification to learn 3D object descriptors from RGB as well as depth images and achieved the best performance in SHREC'15 [21]. Although the classification loss can effectively pull apart objects from different classes thus produce rich inter-class variations in learned feature space, there is little attention attributed to intra-class ones.

In this paper, to deal with both the complex intra-class and inter-class variations, we present a unified multi-view 3D object retrieval approach with a deep embedding network, which refers to the deep convolutional network jointly supervised by classification loss and triplet loss. Since we aim to learn an embedding [22] from the input image space to a discriminative feature space using a deep convolutional network, the learning process of the weight parameters of the network is just the learning process of the embedding. With each triplet consisting of two images from the same class and one from another class, the triplet loss, short for triplet-based ranking loss, aims to separate the positive pair of the triplet from the negative by a distance margin. It was firstly introduced in [23] for nearest-neighbor classification and has been successfully applied to many challenging tasks such as image retrieval [24], [25] and face recognition [20], [26]. Different from the classification loss, triplet loss can not only increase inter-class variations to some extent but also reduce intra-class variations significantly. Nevertheless, we speculate that the attention on inter-class variations from triplet loss is not much enough and classification loss is complementary in this aspect. Besides, classification loss is demonstrated to be quite useful for category-level object retrieval [19]. Therefore, in this paper, both the classification loss and triplet loss are combined together to train the deep embedding network for this category-level multi-view 3D object retrieval. Through the joint optimization of these two kinds of losses and the hierarchical nonlinear mappings of CNN, we can learn an embedding feature space where the Euclidean distance of features directly corresponds to the image semantic similarity, namely, images of the same class are closer than those from different classes. By encoding the relative semantic similarity relationships, we obtain an effective feature space tailored for object retrieval and outperform those deep models trained with classification loss or triplet loss individually.

Additionally, in view of the power of lower layers of CNN in preserving local characteristics and that of higher layers in capturing global semantic information [27], we conduct an extensive comparison for the effectiveness of deep features from different layers of the embedding network. Then, with the deep embedding network, each 3D object is represented with a set of deep features, each of which not only contains abstract semantic information but also captures discriminative local patterns. With different features characterizing varied aspects, the whole feature set forms a comprehensive 3D object representation. Finally, the multi-view 3D object retrieval task is formulated as a set-to-set matching problem, which can be well solved with a variant of Hausdorff distance [8].

The main contributions are summarized as follows:

- We propose a multi-view 3D object retrieval approach based on a deep embedding network jointly supervised by triplet loss and classification loss.
- We perform a wide evaluation on the effectiveness of features from different layers of the deep embedding network. And a compact deep feature which is extracted from the last convolutional layer and characterizes both the discriminative local patterns and global semantic information attains the best performance.
- To train the deep embedding network efficiently, we propose an efficient optimization approach and adopt an effective triplet selection scheme. With the above merits, our proposed approach shows a great advantage over the previous state-of-the-art in SHREC'15, outperforming by over 12%.

The rest of the paper is organized as follows. Section II reviews the related work. Section III describes the architecture of the deep embedding network and the joint optimization of triplet loss and classification loss. Section V presents different feature extraction and set-to-set matching approaches. Section VI conducts detailed retrieval performance comparisons and analysis. Section VII concludes this paper.

II. Related Work

Since our work is clearly in the framework of view-based 3D object retrieval, we will first briefly review some representative works of this field. Early view-based method LFD [12] captured 20 representative views from the vertices of a dodecahedron over a hemisphere and used both Zernike moments and Fourier descriptors to represent a 3D object. This visual similarity-based approach was robust against similarity transformation, noise, model degeneracy etc. Elevation Descriptor (ED) [28] described the altitude information of a 3D object by acquiring six elevations from the front, top, right, rear, bottom, and left views, thus it was invariant to translation and scaling of 3D models. Then Ohbuchi *et al.* [29] utilized SIFT to describe the local characteristics of 3D shapes. In addition, Bag-of-Features [30] was adopted to reduce the computation complexity. Later on, Lian *et al.* [31] employed bag-of-features and clock matching on a set of depth-buffer views obtained from the projections of the normalized object. This method jointly applied the preserved local details and isometry-invariant global structure to reach a competing result. Meanwhile, Papadakis *et al.* [13] proposed PANORAMA, a 3D object descriptor which captured the panoramic view of a 3D shape by projecting it to a lateral surface of a cylinder parallel to one of its three principal axes. Through aligning its principle axes to acquire the global information

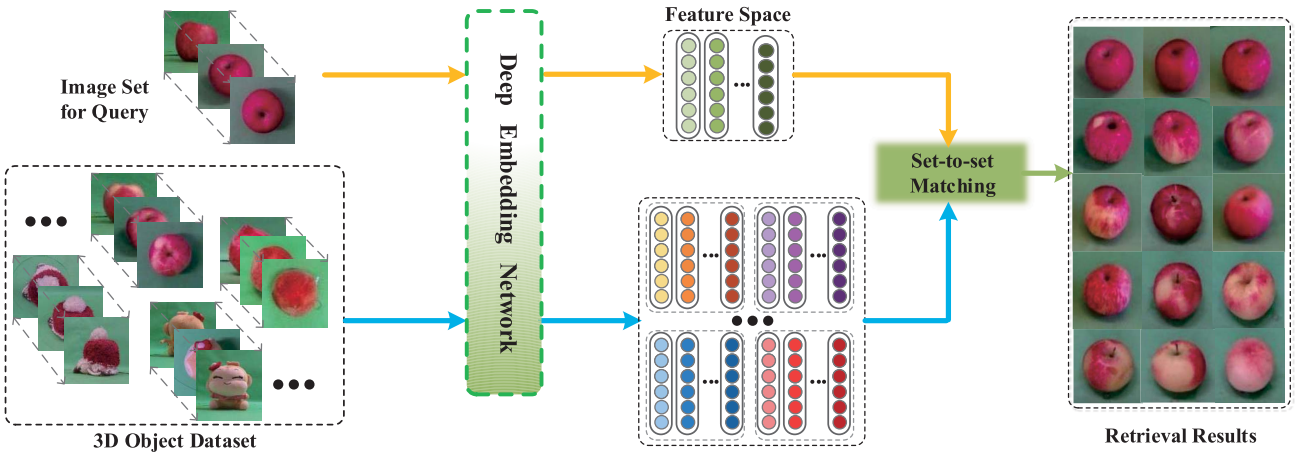


Fig. 1. Overview of the proposed multi-view 3D object retrieval method.

and combing 2D discrete wavelet transform with 2D discrete Fourier transform, PANORAMA achieved record-beating performance on several 3D shape retrieval benchmarks. They also proposed a hybrid shape descriptor in [32] which combined depth buffer based 2D features and spherical harmonies based 3D features. This hybrid method utilized two alignment ways to compensate inner rotation variance and adopted Huffman coding to further compress the hybrid descriptors.

Apart from these traditional methods which take advantage of hand-crafted features as 3D object descriptors, deep neural networks such as auto-encoder and CNN were recently applied to learn more discriminative representation. Zhu *et al.* [8] made use of auto-encoder to learn a 3D shape representation from the projected depth images. By combining the global deep representation and the complementary local descriptor, e.g., SIFT, this approach obtained a high 3D shape retrieval performance. In SHREC'15, to comprehensively characterize a 3D object, we employed three CNN models to learn three kinds of deep CNN-based features separately. Specifically, we utilized a 19-layer CNN model [33] pre-trained on ILSVRC'12 to learn a deep structure feature, which captured rich semantic and structure information. Considering that color is an effective supplement for structure in image retrieval, we took advantage of a deep color CNN model [34] trained for categorizing 10 dominant colors to abstract the deep color feature. Afterwards, we adapted another CNN model [35] pre-trained on RGB images to generate deep features for depth images, which involved shape and distance information of 3D objects. By fusing these three kinds of deep features, we could describe the shape, color, structure and semantic information of a 3D object and finally won the 1st place in the 3D object retrieval contest. Apart from our method, there were several other CNN-based approaches in SHREC'15, which achieved good retrieval performance as well. Although the above CNN-based features are more discriminative than those based on auto-encoders or hand-crafted features, they are tailored for image classification, thus neglecting large intra-class variations.

To cope with the complex intra-class and inter-class variations, apart from classification loss, several similarity constraints, such as contrastive loss [36] and triplet loss, have been proposed for feature learning in computer vision field recently. For example, Sun *et al.* [37] jointly used classification loss and contrastive loss to train CNN models, expecting to develop effective face representations for reducing intra-personal variations while enlarging inter-personal differences. Contrastive loss was firstly proposed in [36] for dimensionality reduction and aimed to minimize the distance between images of the same class while enforce a margin between the distance of images from different classes. In [37], it was mainly used to cover the shortage of classification loss in decreasing large intra-personal variations. Different from the contrastive loss, which only compares pairs of images, triplet loss encourages a relative distance constraint and attracts more research attention. Wang *et al.* [24] presented a deep ranking model consisting of a novel multi-scale network and a triplet loss to characterize the fine-grained visual similarity for images within the same category. Empirical evaluation showed that this deep ranking model outperformed those deep classification models. Later on, also with the help of triplet loss, FaceNet [20] trained a CNN model to learn a mapping from face images to a compact feature space where distances directly reflect face similarity. An online negative exemplar mining strategy was proposed to accelerate CNN convergence and boost the performance. By encoding relative semantic similarity relationships in the learned feature space, triplet loss demonstrated to be more suitable for image ranking than contrastive loss. Whereas, we speculate that, for category-level multi-view 3D object retrieval, it's not sufficient to use triplet loss alone. Considering classification loss could effectively increase inter-class variations and was widely applied for category-level object retrieval, we combine triplet loss with classification loss in deep embedding learning for this retrieval task.

III. Overview

As illustrated in Fig. 1, the proposed approach consists of two stages. In the first stage, for each 3D object with a group of 2D images, a set of deep features are extracted with the deep embedding network. In the second stage, we formulate

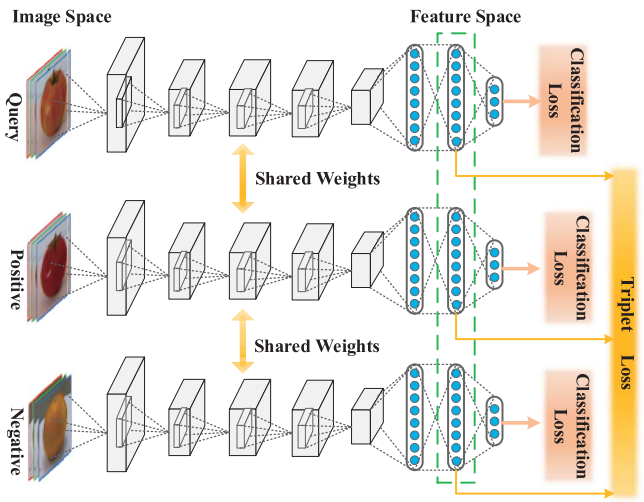


Fig. 2. The training framework of deep embedding network.

the retrieval task as a set-to-set matching problem and the final retrieval results can be obtained by conducting set-to-set matching between the query and all the objects in the retrieval dataset.

IV. Deep Embedding Network

The goal of the embedding process is to learn a mapping function $f(I) = \mathbf{Z}$, transforming the input image I into a point $\mathbf{Z}$ of a feature space where the Euclidean distance of two points directly corresponds to the semantic similarity of input images. To be specific, given two input images I_a and I_b, we define the similarity based on the squared Euclidean distance in the embedding feature space:

$$D(I_a, I_b) = \|f(I_a) - f(I_b)\|_2^2 . \quad (1)$$

The smaller the distance $D(I_a, I_b)$ is, the more similar the two images I_a and I_b are.

A. Network Architecture

Since introduced by LeCun *et al.* [38] in the early 1990's, CNN has shown great power in learning discriminative visual features and achieved record-beating performance at challenging tasks such as large-scale image classification and object detection. Therefore, we take advantage of CNN to model the mapping function $f(I; W_c) = \mathbf{Z}$, where W_c denotes the weight parameters in the deep feature $\mathbf{Z}$ extraction process. Fig. 2 briefly demonstrates the training framework of the deep embedding network. In the training phase, we firstly select triplets, each of which consists of a query, a positive, and a negative image, from the training mini-batch and then use them to train the deep embedding network. The training framework involves three deep embedding networks, sharing the parameters with each other. For each triplet, we compute three classification losses and one triplet loss based on the deep features in the embedding space and then back-propagate the gradients in an optimized way (refer to Algorithm.1).

In terms of the network architecture, we adopt VGG19, a 19-layer very deep CNN presented by Simonyan *et al.* in [33]. Since the specific configurations of VGG19 are detailed out in [33], here we only give a brief description for it. To begin with, the input of VGG19 is a fixed-size 224×224 RGB image, pre-processed by subtracting the mean RGB value, computed on the training set, from each pixel. The image is then passed through a stack of 16 convolutional layers and 3 fully-connected layers. Five max-pooling layers are inserted among convolutional layers to halve the resolution of feature maps. The first two fully-connected layers have 4096 channels each, and the third one performs n-way classification thus has n channels. In the implementation, the classification loss layer follows the last fully-connected layer and the triplet loss layer is added to the second fully-connected layer. As illustrated in Fig. 2, the calculations of the two kinds of losses are independent of each other.

B. Embedding Formulation

To learn the embedding feature space, we jointly use classification loss and triplet loss to supervise the network training. The first loss is to minimize the cross-entropy loss defined as:

$$C(I, t; W_c, W_l) = -log\, \hat{p}_t, \quad (2)$$

where t corresponds to the target class of the input image I and $\hat{p}_t$ is the predicted probability of image I belonging to class t, that is the $t-$th activation in soft-max layer. And the output of soft-max layer can be given by:

$$\mathbf{P} = \delta(W_l f(I; W_c)), \quad (3)$$

where $\delta(.)$ is the soft-max activation function. W_l denotes the parameter weights between the feature extraction layer and the classification loss layer of the deep embedding network. Our goal is to learn the parameters W_c in the feature extraction function $f(I; W_c)$, while W_l is only introduced to propagate the classification loss. To accurately classify all the classes simultaneously, the deep features must have as large inter-class variations as possible.

In contrast, triplet loss regularizes the deep feature by separating the similar pair in the triplet from the dissimilar pair by a distance margin, thus it can reduce the intra-class variations effectively and enlarge the inter-class ones to some extent in the meantime. Given a triplet (I_q, I_p, I_n), where I_q, I_p, I_n are the query image, positive image and negative image respectively. We can define the triplet loss as:

$$T(I_q, I_p, I_n; W_c) = max\{0, m + D(I_q, I_p) - D(I_q, I_n)\}, \quad (4)$$

where m is a margin parameter used to regularize the gap between the distance of two image pairs: (I_q, I_p) and (I_q, I_n). Since CNN can learn to globally scale the embedding proportional to m, slight variation of m may bring little difference to the final performance of the network. However, the relative scale of the margin and the embedding matters in CNN initialization phase. Setting m too large can make CNN training unstable and diverge; setting it too small can slow down the training too much. The hinge loss is a convex approximation to the 0-1 ranking error loss, which measures

the model's violation of the similarity ranking order specified in the triplet. In addition, to speed up the convergence of deep embedding network, we add L2 normalization before calculating triplet loss.

C. Triplet Selection Scheme

One important part for deep embedding network training is triplet selection. Selecting all possible triplets would lead to many triplets easily fulfilling the similarity constraint defined in (4). These triplets do not contribute to loss reduction but still participate in the forward and backward pass through the network, resulting in the slow convergence of network training. So it's crucial to select hard triplets. Additionally, according to [20], there are two ways to generate triplets: online and offline. With the same mini-batch size, online can produce much more triplet samples than the offline way, thus posing large amounts of distance constraints on a small number of images. Therefore, inspired by [25], we adopt the online hard negative mining strategy of using all query-positive pairs in a mini-batch while selecting the top hardest negatives. Specifically, for each pair I_q, I_p, we compute triplet losses with all other negative matches in the mini-batch and select the top K ones with the highest loss values. Then we apply the sum of losses on these K negative matches as the final triplet loss and back-propagate it to update the network parameters. Since the deep feature of each triplet sample is already computed after the forward propagation, the extra computational cost for online hard negative mining is very small.

D. Optimization

Combining classification loss and triplet loss in a linear weighted way, we obtain the overall objective function for the deep embedding network training:

$$\min L(W_c, W_l) = \frac{\alpha}{2}\cdot\|W_c\|_F^2+\frac{\alpha}{2}\cdot\|W_l\|_F^2+\sum_{x=1}^{M}C(I_x, t_x; W_c, W_l) + \beta\cdot\sum_{i=1}^{N}T(I_{q_i}, I_{p_i}, I_{n_i}; W_c), \tag{5}$$

where M denotes the number of images in a mini-batch and N is the number of triplets selected from the mini-batch using online hard negative mining strategy. With the online triplet generation scheme, N can be much larger than M. I_{q_i}, I_{p_i} and I_{n_i} denotes the query, positive and negative image of the $i-$th triplet. α is the weight decay and β is a hyper-parameter used to balance the classification loss and triplet loss. When $\beta = 0$, the triplet loss vanishes and only the classification loss takes effect. With the increase of β, the triplet loss gradually dominates the training process. When $\beta \to +\infty$, there is only triplet loss remaining. In the experiments, we will show that neither the classification loss nor the triplet loss is the optimal one to learn embedding feature space. Only by combing these two losses with a proper β, can we reach the best result.

According to Fig. 2, the number of network propagations depends on the number of triplets selected from each mini-batch, with each triplet involving three rounds of forward and backward propagation during the network training. However, when the above triplet selection scheme is adopted, there is a very good chance that the same image in a mini-batch occurs in different triplets, thus the forward and backward propagation of that image can be reused. Taking this into consideration, we employ an optimized learning algorithm, which is similar to the image-based gradient descent algorithm proposed in [39], to make the number of network propagations depend only on the number of images in each mini-batch.

Stochastic gradient descent is used to minimize the objective function. From (5), we can get the derivative of the training loss L with respect to the weight parameter W_l:

$$\frac{\partial L}{\partial W_l} = \alpha\cdot W_l + \sum_{x=1}^{M}\frac{\partial C(I_x, t_x; W_c, W_l)}{\partial W_l}, \tag{6}$$

which can be easily obtained by aggregating the derivative of each classification loss with respect to W_l. Then the derivative of the loss L with respect to the weight parameter W_c can be represented as:

$$\frac{\partial L}{\partial W_c} = \alpha\cdot W_c + \sum_{x=1}^{M}\frac{\partial C_x}{\partial W_c} + \beta\cdot\sum_{i=1}^{N}\frac{\partial T_i}{\partial W_c}, \tag{7}$$

where C_x and T_i denotes each classification loss and triplet loss respectively. According to the chain rule in network gradient back-propagation, the derivative of each classification loss with respect to W_c can be given by:

$$\frac{\partial C_x}{\partial W_c} = \frac{\partial C(I_x, t_x; W_c, W_l)}{\partial \mathbf{Z}_x}\frac{\partial \mathbf{Z}_x}{\partial W_c}, \tag{8}$$

where $\mathbf{Z}_x = f(I_x; W_c)$ is the deep feature for the image I_x. Similarly, the derivative of each triplet loss with respect to W_c can be calculated by:

$$\frac{\partial T_i}{\partial W_c} = \sum_{x\in\{q_i, p_i, n_i\}}\frac{\partial T_i}{\partial \mathbf{Z}_x}\frac{\partial \mathbf{Z}_x}{\partial W_c}. \tag{9}$$

To reuse the network forward and backward propagation for some images which occur in multiple triplets, we can rewrite the third part of (7) as:

$$\sum_{i=1}^{N}\frac{\partial T_i}{\partial W_c} = \sum_{x=1}^{M}(\sum_{i=1}^{N}\frac{\partial T_i}{\partial \mathbf{Z}_x})\frac{\partial \mathbf{Z}_x}{\partial W_c}, \tag{10}$$

The form of $\frac{\partial T_i}{\partial \mathbf{Z}_x}$ is up to the role I_x plays in T_i, and, with $\mathbf{N}_x$ denoting the normalized $\mathbf{Z}_x$, we have:

$$\frac{\partial T_i}{\partial \mathbf{Z}_{q_i}} = \lambda_i\cdot 2(\mathbf{N}_{n_i} - \mathbf{N}_{p_i})\cdot\frac{\partial \mathbf{N}_{q_i}}{\partial \mathbf{Z}_{q_i}}, \tag{11}$$

$$\frac{\partial T_i}{\partial \mathbf{Z}_{p_i}} = \lambda_i\cdot 2(\mathbf{N}_{p_i} - \mathbf{N}_{q_i})\cdot\frac{\partial \mathbf{N}_{p_i}}{\partial \mathbf{Z}_{p_i}}, \tag{12}$$

$$\frac{\partial T_i}{\partial \mathbf{Z}_{n_i}} = \lambda_i\cdot 2(\mathbf{N}_{q_i} - \mathbf{N}_{n_i})\cdot\frac{\partial \mathbf{N}_{n_i}}{\partial \mathbf{Z}_{n_i}}. \tag{13}$$

λ_i is a binary parameter indicating whether the triplet loss T_i is zero. Generally, in a mini-batch, only several T_i contain I_x, so it needs little additional computation cost to calculate $\sum_{i=1}^{N}\frac{\partial T_i}{\partial \mathbf{Z}_x}$. Combining (8) and (10), (7) becomes:

$$\frac{\partial L}{\partial W_c} = \alpha\cdot W_c + \sum_{x=1}^{M}(\frac{\partial C_x}{\partial \mathbf{Z}_x} + \beta\sum_{i=1}^{N}\frac{\partial T_i}{\partial \mathbf{Z}_x})\frac{\partial \mathbf{Z}_x}{\partial W_c}. \tag{14}$$

Algorithm 1 Deep Embedding Network Learning Algorithm

Input:
training set $\{(I_x, t_x)\}$, initialized parameters W_c, W_l, hyper-parameters α, β, m, learning rate $\eta(t)$
Output:
network parameter W_c

$t \leftarrow 0$
$\frac{\partial L}{\partial W_c} = 0, \frac{\partial L}{\partial W_l} = 0$;
while $t < T$ **do**
  $t \leftarrow t + 1$;
  Calculate deep feature $\mathbf{Z}_x$ and classification loss C_x by forward propagation;
  Sample triplet set $\{(I_{q_i}, I_{p_i}, I_{n_i})\}$ from the mini-batch with hard negative mining strategy;
  Calculate the derivative $\frac{\partial C_x}{\partial \mathbf{Z}_x}$ and $\sum_{i=1}^{N} \frac{\partial T_i}{\partial \mathbf{Z}_x}$ with (2), (11), (12) and (13);
  Calculate $\frac{\partial C_x}{\partial W_l}$ and $\frac{\partial \mathbf{Z}_x}{\partial W_c}$ by backward propagation;
  Calculate the derivative $\frac{\partial L}{\partial W_l}$ and $\frac{\partial L}{\partial W_c}$ according to (6) and (14);
  Update $W_c = W_c - \eta(t) \cdot \frac{\partial L}{\partial W_c}, W_l = W_l - \eta(t) \cdot \frac{\partial L}{\partial W_l}$.
end while

Taken together, the whole training process of deep embedding network is given in Algorithm.1. Since our learning algorithm is well compatible with Caffe [40], a widely applied deep learning framework, we can implement it by simply modifying the loss layer. By reducing the number of network propagations to the mini-batch size, we can accelerate the network training greatly, demonstrating a crucial advantage for large-scale dataset.

V. Multi-View 3D Object Retrieval

A. Feature Extraction

Generally, CNN-based deep features are extracted from the last fully-connected layers [18], [20], [41]. Although this kind of deep feature captures much semantic information and has obtained promising results for several visual recognition tasks, it lacks enough local characteristics of objects, which are quite crucial for multi-view 3D object retrieval. It has been demonstrated [42], [27] that the feature from lower layers of CNN can extract more local patterns of objects and that from higher layers is inclined to capture more global abstract information. The deep feature suitable for multi-view 3D object retrieval task should be a tradeoff between local characteristics and global semantic information. Taking this into account, we extract deep features from different layers of the deep embedding network and evaluate their effectiveness through extensive retrieval experiments. Specifically, for the fully-connected layers, we directly extract deep features from the layer output. In terms of convolutional layers, we extract deep features by aggregating local activations in the feature map stack. Each activation corresponds to a local region of the input image and captures some local patterns of this region. We explore several ways to aggregate these local convolutional activations, including max-pooling, average-pooling, and concatenation. Max-pooling is to first find the greatest activation in each feature map and then concatenate them to form the final deep feature. It only focuses on the discriminative local characteristics of the image. Different from it, average-pooling is to calculate the average of all the activations in each feature map and considers all the local patterns comprehensively. Concatenation is to directly concatenate all the local activations in the convolutional layer thus takes all the local patterns of the input image into account. Usually, deep features aggregated with the above two pooling strategies have lower dimensionality than those extracted from fully-connected layers, whereas deep features aggregated by concatenation are rather high-dimensional.

B. Set-to-Set Matching

With the deep embedding network, each 3D object can be represented by a set of deep features. Thus the original 3D object retrieval task is converted to a set-to-set matching problem, which can be well solved via set-to-set distances. In this paper, we evaluate three set-to-set distances: minimum distance we used in SHREC'15, Hausdorff distance [43], and a variant of Hausdorff distance [8]. With O_A, O_B denoting two 3D objects, minimum distance defines their distance as:

$$H(O_A, O_B) = min_{a \in O_A} min_{b \in O_B} D(I_a, I_b), \tag{15}$$

where I_a and I_b denotes the $a-$th image O_A has and the $b-$th image O_B has respectively. D is the distance function given in (1). This distance takes the minimum image-to-image distance as the final set-to-set distance. Hausdorff distance is defined as:

$$H(O_A, O_B) = max_{a \in O_A} min_{b \in O_B} D(I_a, I_b). \tag{16}$$

By adopting a max-min scheme, Hausdorff distance is able to explore the closest matching for each view and has shown its effectiveness for set-to-set matching [44]. The third distance is a modified version of Hausdorff distance, which is given by:

$$H(O_A, O_B) = \frac{1}{C} \sum_{a \in O_A} min_{b \in O_B} D(I_a, I_b), \tag{17}$$

where C represents the number of images each 3D object has. Different from the above two distances, it takes the whole view set into account.

VI. Experiments

A. Datasets

To evaluate the performance of our method, two large-scale multi-view object datasets, i.e., RGB-D Object Dataset [45] and MV-RED [21], have been employed in our experiments. RGB-D Object Dataset is used to train the deep embedding network and MV-RED is used for evaluation. It is worth noting that these two datasets share some object categories, which ensures the CNN model trained on one dataset can transfer well to the other one. For all objects in both of the two datasets, multi-view RGB and depth images are captured at 640×480 resolution by RGB-D cameras.

1) RGB-D Object Dataset: this dataset contains 250,000 RGB-D images of 300 common everyday objects from multiple views. Each object belongs to one of 51 categories from ImageNet [41] and is represented with RGB and depth images from roughly 750 views. This dataset also provides segmentation masks, using which we can get the bounding box for each object.

2) MV-RED: this real world 3D object dataset with multimodal views is collected for SHREC'15. It consists of 505 objects, which can be divided into 60 categories. The complete version of MV-RED, named as MV-RED-721, characterizes each 3D object with RGB-D images captured from 721 views. Uniformly sampling the 721 views with the step of 10 degrees, we obtain a compact version of MV-RED, referred to as MV-RED-73. Foreground segmentation results for RGB images are provided too.

B. Evaluation Protocol

Following the practice of SHREC'15, we select the categories involving more than 10 objects in MV-RED as the query set, leading to 311 queries in total. In terms of the retrieval set, it involves all the 505 objects in MV-RED. Similar to SHREC'15, we conducted two retrieval experiments on MV-RED-721 and MV-RED-73 respectively. Seven evaluation criteria are used to evaluate the performance of different multi-view 3D object retrieval methods. In our experiments, for each of the retrieval results, if it belongs to the same category as the query object, we regard it as the relevant object. For the calculation of each evaluation criterion, we average it over 311 queries.

1) *Precision-Recall Curve* is assessed in terms of average recall and precision and it intuitively shows the retrieval performance.
2) *Nearest Neighbor (NN)* evaluates the retrieval accuracy of the second returned result, since the first returned one in our experiments must be the query itself.
3) *First Tier (FT)* denotes the recall of the top τ results, with τ defined as the the number of relevant objects in the whole dataset for the query.
4) *Second Tier (ST)* denotes the recall of the top 2τ results.
5) *F-Measure (F)* jointly evaluates the precision and recall for a fixed number of top returned results. In our experiments, the top 20 retrieved results are considered.
6) *Normalized Discounted Cumulative Gain (NDCG)* is a statistic that assigns relevant results at top ranking positions with higher weights, under the assumption that a user is less likely to consider lower results.
7) *Average Normalized Modified Retrieval Rank (ANMRR)* is a rank-based measure considering the ranking information of relevant objects among the returned results. Different from the above measures, a lower ANMRR value indicates a better performance.

C. Implementation Details

In the training phase, we use all the cropped RGB images in RGB-D Object Dataset as the training data and resize them to 224×224 size firstly before sending into the network. We initialize the deep embedding network with VGG19 model pre-trained on ImageNet and the weight decay α is fixed to 0.0005. As for the batch size, the more larger it is, the more triplets will be formed in a mini-batch. Then we have a better chance of selecting more and harder triplets in each mini-batch. When we set the batch size to the size of the whole training set, we are actually mining hard triplets from all the training images, which is just what we aim to approximate with the above mini-batch based hard triplets mining strategy. However, large batch size leads to high GPU memory consumption when training CNN. Hence we set batch size to 100 when we train the deep embedding network on a GTX TITAN X GPU with 12G memory. We adopt a fixed learning rate of 1e-5 during the network training. In our experiments, we find that the final retrieval performance is not sensitive to the value of margin m when it varies between 0.1 and 0.8. Thus when using L2 normalization, we set $m = 0.2$. When selecting hard triplets online, we set K to 30 for the mini-batch size of 100. Too small K will pose insufficient distance constraints on each image, whereas too large K will violate the hard triplet selection scheme. Regarding the hyper-parameter β, we empirically fix it to 0.01. We find that small variations to β, e.g., from 0.01 to 0.1, bring little difference to the final retrieval performance. For 3D object retrieval both on MV-RED-73 and MV-RED-721, we firstly use the foreground segmentation mask to crop the object region before inputting to CNN to extract deep features.

TABLE I

NETWORK PROPAGATION TIME COMPARISON OF DIFFERENT OPTIMIZATION APPROACHES

Approach	Batch=20	Batch=50	Batch=100
Ours	0.417s	1.043s	1.97s
Orig	10.549s	72.966s	307.241s

D. Comparison of Different Optimization Approaches

Table I demonstrates the time it takes to forward and backward propagate the whole mini-batch through the deep embedding network for once during the network training, under the different batch size. When we adopt the method "Orig," the number of network propagations for one mini-batch depends on the number of triplets generated in the mini-batch, with each triplet involving three rounds of forward and backward propagation. "Ours" denotes the optimization algorithm we proposed in this paper, with which the number of network propagations for one mini-batch reduces to the batch size. For example, if we set the batch size to 20, let us suppose there are 10 query-positive pairs, then there will be at most 180 triplets generated. Then according to "Ours," there will be 20 rounds of propagation for this mini-batch, while according to "Orig," there will be $180 \times 3 = 540$ rounds of propagation. Theoretically, the larger the mini-batch is, the more triplets will be generated and the more time will be saved for the propagation of this mini-batch using the proposed optimization approach. All the time in Table I is calculated on

TABLE II
RETRIEVAL PERFORMANCE COMPARISON OF DIFFERENT DEEP FEATURES

Feature	DIM	NN	FT	ST	F	NDCG	ANMRR
FC7	4096	0.91	**0.5758**	0.7121	0.5756	0.6927	**0.3394**
FC6	4096	0.8939	0.5698	0.7122	0.5714	0.6897	0.3446
Conv5-4(ave)	512	0.8971	0.5654	0.7002	0.5655	0.6837	0.3485
Conv5-4(cas)	25088	0.9132	0.5658	0.7045	0.5651	0.6846	0.3492
Conv5-4(max)	512	**0.9228**	0.5736	**0.722**	**0.5811**	**0.6934**	0.3397
Conv5-3	512	0.9228	0.5673	0.7099	0.5723	0.6854	0.3483
Conv5-2	512	0.9132	0.5661	0.711	0.5716	0.6831	0.3492
Conv5-1	512	0.8939	0.5568	0.6953	0.5597	0.6731	0.3583
Conv4-4	512	0.9068	0.5399	0.6776	0.5415	0.6553	0.3765
Conv4-3	512	0.8585	0.4951	0.6113	0.4941	0.6062	0.4203

a GTX TITAN X GPU and the comparison results demonstrate the efficiency of the proposed optimization algorithm well.

E. Evaluation on Different Deep Features

Table II shows the comparison experimental results of deep features extracted from different layers of deep embedding network. "FC6" and "FC7" denotes the 4096-dimensional deep feature extracted from the first and second fully-connected layer respectively, while the remaining features are all from convolutional layers. "Conv5-4(ave)," "Conv5-4(cas)," and "Conv5-4(max)" is the deep feature obtained by aggregating local activations of the last convolutional layer "Conv5-4" with average-pooling, concatenation, and max-pooling separately. Both "Conv5-4(ave)" and "Conv5-4(max)" are 512-dimensional, whereas "Conv5-4(cas)" is 25088-dimensional. From Table II we can see that "Conv5-4(max)" beats "Conv5-4(ave)" and "Conv5-4(cas)" on all evaluation criteria, which shows that focusing on the discriminative local patterns rather than taking all into consideration indiscriminatively is more effective for multi-view 3D object retrieval. The remaining convolutional deep features in Table II are all aggregated with max-pooling strategy and 512-dimensional. On one hand, it can be found that the performance of deep features obtained from high to low convolutional layers reduces gradually. For example, compared to "Conv5-4(max)," "Conv4-3" decreases greatly by around 8%, which may result from the decrease of semantic information. On the other hand, although "FC6" and "FC7" involves much semantic information, they are inferior to "Conv5-4(max)" on most evaluation indexes, which is probably due to the lack of sufficient discriminative local characteristics. Therefore, it can be verified that an effective deep feature for multi-view 3D object retrieval task should be a tradeoff between global semantic information and discriminative local patterns. The performance comparison is conducted on MV-RED-73 and the modified Hausdorff distance introduced above is adopted for set-to-set matching. Besides, "Conv5-4(max)" is more compact than fully-connected features, reducing the computational cost greatly. Therefore, we adopt it in the following retrieval experiments.

TABLE III
RETRIEVAL PERFORMANCE COMPARISON OF DIFFERENT SET-TO-SET DISTANCES

Distance	NN	FT	ST	F	NDCG	ANMRR
Hausd(O)	0.9068	0.533	0.6799	0.5409	0.6492	0.3829
Min	0.8875	0.5368	0.679	0.5502	0.6538	0.3753
Hausd(M)	**0.9228**	**0.5736**	**0.722**	**0.5811**	**0.6934**	**0.3397**

TABLE IV
COMPARISON WITH ALL THE METHODS ON MV-RED-73

Method	NN	FT	ST	F	NDCG	ANMRR
GMM-Zernike	0.5177	0.2561	0.3630	0.2784	0.3239	0.6867
GMM-HoG	0.8264	0.3594	0.4261	0.3662	0.4856	0.5476
BGM-Color	0.5756	0.2728	0.3905	0.3188	0.3417	0.6427
BGM-HoG	0.7717	0.3536	0.4254	0.3582	0.4621	0.5586
XMU-GS	0.8006	0.3735	0.4444	0.3701	0.482	0.5407
XMU-GS-FB	**0.9453**	0.4533	0.5346	0.4467	0.5977	0.4538
BTBU-BoF	0.4502	0.2493	0.3586	0.2987	0.3051	0.6653
BTBU-MVM	0.6109	0.2661	0.3766	0.3156	0.3397	0.6456
CAS-ECKM	0.8939	0.4034	0.4752	0.4016	0.5368	0.5042
CAS-ECR	0.9196	0.4156	0.5036	0.4202	0.5582	0.4895
CAS-CSR	0.9196	0.435	0.5271	0.4355	0.5763	0.4713
IVA-DeepColor	0.7556	0.4663	0.592	0.4632	0.554	0.4636
IVA-Deep4	0.7717	0.4753	0.5831	0.4636	0.5689	0.4534
DeepEm(C-FC7)	0.9196	0.5674	0.6976	0.5672	0.6868	0.3458
DeepEm(T-FC7)	0.91	0.5704	0.7109	0.5671	0.6881	0.3453
DeepEm(M-FC7)	0.91	**0.5758**	0.7121	0.5756	0.6927	**0.3394**
DeepEm(C-FC6)	0.9132	0.5621	0.7007	0.5623	0.6844	0.3508
DeepEm(T-FC6)	0.8907	0.5645	0.7058	0.5662	0.6841	0.3509
DeepEm(M-FC6)	0.8939	0.5698	0.7122	0.5714	0.6897	0.3446
DeepEm(C-Conv5-4)	0.91	0.5655	0.7143	0.578	0.6858	0.3479
DeepEm(T-Conv5-4)	0.926	0.5678	0.7173	0.5786	0.689	0.345
DeepEm(M-Conv5-4)	0.9228	0.5736	**0.722**	**0.5811**	**0.6934**	0.3397
DeepEm(C-Conv5-3)	0.9228	0.5518	0.671	0.5452	0.6725	0.3621
DeepEm(T-Conv5-3)	0.91	0.5532	0.6794	0.5483	0.6245	0.361
DeepEm(M-Conv5-3)	0.9228	0.5673	0.7099	0.5723	0.6854	0.3483
DeepEm(C-Conv5-2)	0.9164	0.5352	0.6483	0.5217	0.6587	0.3784
DeepEm(T-Conv5-2)	0.9196	0.5332	0.6509	0.5259	0.6577	0.3804
DeepEm(M-Conv5-2)	0.9132	0.5661	0.711	0.5716	0.6831	0.3492
DeepEm(C-Conv5-1)	0.9003	0.517	0.6269	0.5075	0.6411	0.3968
DeepEm(T-Conv5-1)	0.9068	0.5156	0.6294	0.5075	0.6403	0.3976
DeepEm(M-Conv5-1)	0.8939	0.5568	0.6953	0.5597	0.6731	0.3583

F. Evaluation on Different Set-to-Set Distances

Table III illustrates the evaluation results on three set-to-set distances we mentioned above. "Hausd(O)" represents the original Hausdorff distance and "Min" denotes the distance we used in SHREC'15, both of which select only one image-to-image distance as the final set-to-set distance and thus are likely to be affected by noise and view-point change. "Hausd(M)" denotes the modified Hausdorff distance, which takes all the views into consideration and comprehensively measures the distance between two 3D objects. This comparison experiment is also conducted on MV-RED-73 and "Conv5-4(max)" in Table II is used as the deep feature. As shown in Table III, "Hausd(M)" demonstrates an apparent advantage over others, verifying its effectiveness for multi-view 3D object retrieval. We finally employ it for set-to-set matching.

G. Comparison to State-of-the-Art

We compare our approach with the state-of-the-art methods in SHREC'15 [21] and Table IV illustrates the results

on MV-RED-73. The top eight methods in Table IV all utilize hand-crafted features, like Zernike moment, HoG, Fourier descriptor, etc. Both "GMM-Zernike" and "GMM-HoG" firstly group all views into clusters, and then only use representative views selected from the clusters for retrieval. Specifically, they use Zernike moment extracted from RGB image and HoG from depth image as the feature for clustering respectively, and a Gaussian model is learned to model the feature distribution in each cluster. Similar to them, "BGM-Color" and "BGM-HoG" also choose to cluster all the views to reduce redundant information, with both visual and spatial information of 2D images. Graph cut is used to get a set of sub-clusters and 3D objects are compared with bipartite matching. The main idea of "XMU-GS" is to formulate the relationship between two 3D objects with three bipartite graphs, which are constructed using Zernike moment, Fourier descriptor and BoWs respectively. "XMU-GS-FB" introduces user relevance feedback information to "XMU-GS." In "BTBU-BoF" and "BTBU-MVM," a hybrid shape descriptor ZFCE, consists of Zernike moment, Fourier feature, circularity feature and eccentricity feature, is built and Bag-of-Feature (BoF) and multiple view matching (MVM) is adopted to compute the similarity between 3D objects separately.

The following five methods in Table IV are all deep learning-based methods and they demonstrate a clear advantage over all the traditional methods, except "XMU-GS-FB," which makes use of user relevance feedback information in retrieval process. This indicates that deep learning methods are able to explore more discriminative features for 3D objects. To comprehensively describe each 3D object, "CAS-ECKM," "CAS-ECR" and "CAS-CSR" leverage four modalities, i.e., RGB, gray-scale, depth and surface normal, to learn the visual appearance and shape cues of 3D object. In feature learning process, "CAS-ECKM" adapts a single-layer network based on K-means clustering for 2D images, "CAS-ECR" combines a single CNN and multiple RNNs, and "CAS-CSR" employs SPM to adapt "CAS-ECR" for images of arbitrary sizes. Two linear SVM classifiers are trained with the above features and then fused to predict the category of each query.

In our previous methods in SHREC'15 , "IVA-DeepColor" and "IVA-Deep4," we only leverage CNN models pre-trained on external data to learn 3D object representations. To be specific, we use VGG19 pre-trained on ImageNet to extract two deep structure features and adopt color CNN model trained for categorizing 10 dominant colors to abstract deep color feature. All these three features are extracted from RGB images and form the 3D descriptor used in "IVA-DeepColor." Apart from the above features from RGB images, "IVA-Deep4" also utilizes deep depth feature extracted from transformed depth image with a 8-layer CNN model pre-trained on ImageNet [35]. Specifically, to learn effective deep features from depth images, which involve the shape and distance information of 3D objects, through CNN, we firstly transform depth images into RGB images with HHA, a kind of encoding strategy proposed in [46]. The three channels of the produced RGB images represent the height above ground, horizontal disparity and the pixelwise angle between a surface normal and the gravity direction separately. Then we tried two CNN models to extract deep features from the produced RGB images. One model is the CNN model [35] pre-trained on ImageNet, the other is trained by fine-tuning the second CNN model on RGB images transformed from all the depth images in RGB-D Object Dataset. However, deep feature extracted from the second CNN model performed rather poor, just slightly better than that from the first CNN model. As shown in Table IV, it seems that the use of deep depth features does not bring performance improvement, since there is no considerable difference between "IVA-DeepColor" and "IVA-Deep4." The possible reasons for this are two-fold. On the one hand, the depth information delivered in each depth image is not sufficient, since the object within the depth images captured from various angles looks relatively small. On the other hand, the depth information is not discriminative enough to separate some objects of the similar shape, such as "Apple" and "Orange." Considering our unsuccessful use of the depth modality for CNN training in SHREC'15, we do not use the depth images to train CNN models in this paper. Nevertheless, we will try to deal with it in our future work. From Table IV we can see that CNN models pre-trained on ImageNet for image classification can transfer well for view-based 3D object retrieval on MV-RED, outperforming the remaining deep-based methods on four out of six evaluation criteria. Comparison experiments on MV-RED-721 also show the advantage of our methods and the results are listed in Table V.

TABLE V
COMPARISON WITH ALL THE METHODS ON MV-RED-721

Method	NN	FT	ST	F	NDCG	ANMRR
GMM-Zernike	0.4534	0.2225	0.318	0.2505	0.2804	0.7228
GMM-HoG	0.7524	0.3697	0.4404	0.3688	0.4783	0.5452
BGM-Color	0.5305	0.2669	0.3865	0.315	0.3267	0.6484
BGM-HoG	0.7428	0.3477	0.4101	0.3514	0.4562	0.5643
XMU-GS	0.7074	0.3199	0.4129	0.3375	0.4033	0.5983
XMU-GS-FB	0.9132	0.4356	0.5403	0.4427	0.5724	0.4717
BTBU-BoF	0.5273	0.2394	0.3443	0.2945	0.3019	0.6728
BTBU-MVM	0.6109	0.2720	0.3688	0.3146	0.3494	0.6413
CAS-ECKM	0.8939	0.4034	0.4752	0.4016	0.5368	0.5042
CAS-ECR	0.9196	0.4156	0.5036	0.4202	0.5582	0.4895
CAS-CSR	0.9196	0.435	0.5271	0.4355	0.5763	0.4713
IVA-DeepColor	0.7492	0.4671	0.5912	0.4641	0.5537	0.4629
IVA-Deep4	0.7588	0.4615	0.5777	0.4531	0.5536	0.4675
DeepEm(M)	**0.9228**	**0.5749**	**0.7144**	**0.5797**	**0.6939**	**0.3388**

The rest of Table IV are all results from three deep CNN models we trained for this retrieval task, i.e., DeepEm(C), DeepEm(T) and DeepEm(M). All the three CNN models are trained by leveraging RGB-D Object Dataset to fine-tune VGG19 model pre-trained on ImageNet. The difference lies in the supervisory signal: DeepEm(C) and DeepEm(T) makes use of classification loss and triplet loss individually, whereas DeepEm(M) utilizes the combination of these two losses. Both "DeepEm(C-FC7)" and "DeepEm(C-FC6)" adopt the deep features extracted from fully-connected layers. "DeepEm(C-Conv5-4)," "DeepEm (C-Conv5-3)," "DeepEm(C-Conv5-2)" and "DeepEm (C-Conv5-1)" all utilize the deep feature learnt from

convolutional layers, which is aggregated with max-pooling strategy. Considering NN is not a comprehensive evaluation index of retrieval performance, we pay more attention to other evaluation criteria when comparing the results of different methods. From Table IV, we can see a consistent advantage of DeepEm(M) over DeepEm(C) and DeepEm(T), whatever deep feature we use, which can well demonstrate the superiority of our proposed method. Specifically, from the results of "DeepEm(M-FC7)," "DeepEm(M-FC6)" and "DeepEm (M-Conv5-4)," we can notice 0.5% average performance improvement for all the evaluation criteria except NN. And the average improvement for "DeepEm(M-Conv5-3)," "DeepEm(M-Conv5-2)" and "DeepEm(M-Conv5-1)" is 1.8%, 3.9% and 4.6% separately. It can be seen that although the performance improvements for "DeepEm(M-FC7)," "DeepEm(M-FC6)" and "DeepEm(M-Conv5-4)" are not that significant, those for "DeepEm(M-Conv5-3)," "DeepEm (M-Conv5-2)" and "DeepEm(M-Conv5-1)" are relatively considerable. The reason may be that, for deep features Conv5-4, FC6 and FC7, their discriminative power is already situated in a high level, then the space for performance improvement is relatively limited. However, for deep features extracted from lower layers, a slight change of discriminative power resulted from the change of training method may lead to a considerable performance difference.

Additionally, as illustrated in Table IV, both DeepEm(C) and DeepEm(T) outperform the state-of-the-art in SHREC'15 by a large margin, indicating the effectiveness of classification loss and triplet loss for category-level multi-view 3D object retrieval. And DeepEm(T) performs slightly better than DeepEm(C), which may result from the extra attention on intra-class variations from triplet loss. DeepEm(M) boosts the retrieval performance further by at least 0.5% on average, which verifies the complementarity between these two losses. Therefore, by fusing classification loss and triplet loss properly, we can learn an embedding space where inter-class variations are large enough while intra-class ones are effectively reduced. As illustrated in Table IV, "DeepEm (M-Conv5-4)" achieves the best retrieval performance, beating the state-of-the-art by a large margin. Additionally, the results of the proposed method also demonstrates the great generality of the CNN model trained on RGB-D Object Datset to the multi-view 3D object retrieval task on MV-RED.

Our proposed approach also demonstrates its advantage on MV-RED-721, as shown in Table V, where "DeepEm(M)" denotes the result with deep feature "Conv5-4." Besides, we find an interesting phenomenon that using 721 views to describe each 3D object is not necessarily better than 73 views, namely, the results using 721 views do not have significant improvement than that using 73 views for almost all methods and the performance is even degraded when employing more views for some methods. We speculate that 73 views have effectively captured the 3D local structure properties for 3D objects and the abundant information involved in 721 views interferes with the efficient 3D object representation learning. Fig. 3 and Fig. 4 illustrates the Precision-Recall Curve comparison for different methods on MV-RED-73 and MV-RED-721 separately, which demonstrates the superiority

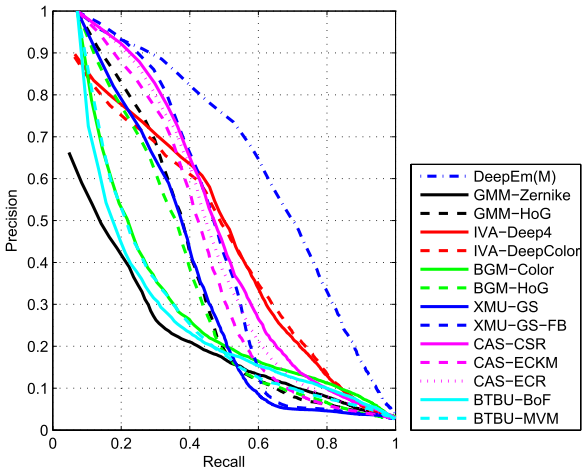


Fig. 3. Precision-recall curve for different methods on MV-RED-73.

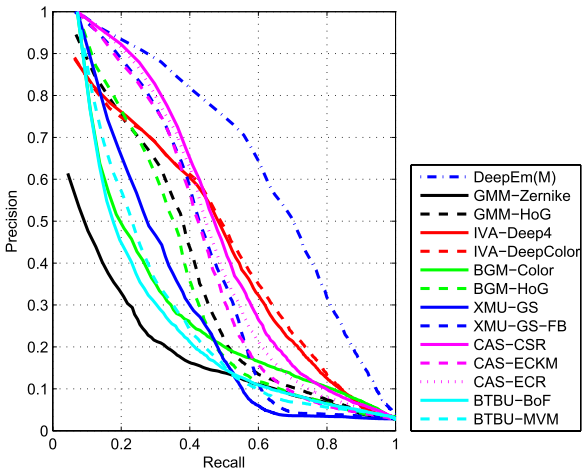


Fig. 4. Precision-recall curve for different methods on MV-RED-721.

of the proposed method again. "DeepEm(M)" in both of Fig. 3 and Fig. 4 refers to the method using the deep feature "Conv5-4."

VII. Conclusion

In this paper, we propose a deep embedding network jointly supervised by multiple losses for multi-view 3D object retrieval. By decreasing the intra-class variations while increasing the inter-class ones, this embedding network can learn a feature space tailored for the retrieval task. With the embedding network, we can extract a set of compact and discriminative deep features and transform the multi-view 3D object retrieval into a set-to-set matching problem. Experimental results show that the proposed approach outperforms the state-of-the-art on MV-RED-73 and MV-RED-721 by a large margin.

References

[1] H.-S. Wong, B. Ma, Z. Yu, P. F. Yeung, and H. H. Ip, "3-D head model retrieval using a single face view query," *IEEE Trans. Multimedia*, vol. 9, no. 5, pp. 1026–1036, Aug. 2007.

[2] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas, "Joint embeddings of shapes and images via cnn image purification," *ACM Trans. Graph.*, vol. 34, no. 6, p. 234, Nov. 2015.

[3] J.-S. Yeh, D.-Y. Chen, B.-Y. Chen, and M. Ouhyoung, "A Web-based Three-Dimensional protein retrieval system by matching visual similarity," *Bioinformatics*, vol. 21, no. 13, pp. 3056–3057, Apr. 2005.

[4] J. W. Tangelder and R. C. Veltkamp, "A survey of content based 3D shape retrieval methods," *Multimedia Tools Appl.*, vol. 39, no. 3, pp. 441–471, Sep. 2008.

[5] Y. Liu, X.-L. Wang, H.-Y. Wang, H. Zha, and H. Qin, "Learning robust similarity measures for 3D partial shape retrieval," *Int. J. Comput. Vis.*, vol. 89, no. 2, pp. 408–431, Sep. 2010.

[6] Y. Gao, Q. Dai, and N.-Y. Zhang, "3D model comparison using spatial structure circular descriptor," *Pattern Recognit.*, vol. 43, no. 3, pp. 1142–1151, Mar. 2010.

[7] P. Daras and A. Axenopoulos, "A 3D shape retrieval framework supporting multimodal queries," *Int. J. Comput. Vis.*, vol. 89, nos. 2–3, pp. 229–247, 2010.

[8] Z. Zhu, X. Wang, S. Bai, C. Yao, and X. Bai, "Deep learning representation using autoencoder for 3D shape retrieval," in *Proc. SPAC*, Oct. 2014, pp. 279–284.

[9] Y. Gao and Q. Dai, "View-based 3-d object retrieval: Challenges and approaches," *IEEE Multimedia Magazine*, vol. 21, no. 3, pp. 52–57, Mar. 2014.

[10] T. F. Ansary, M. Daoudi, and J.-P. Vandeborre, "A Bayesian 3-D search engine using adaptive views clustering," *IEEE Trans. Multimedia*, vol. 9, no. 1, pp. 78–88, Jan. 2007.

[11] Y. Gao *et al.*, "Camera constraint-free view-based 3-D object retrieval," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 2269–2281, Apr. 2012.

[12] 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.

[13] 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.

[14] W.-Y. Kim and Y.-S. Kim, "A region-based shape descriptor using Zernike moments," *Signal Process., Image Commun.*, vol. 16, pp. 95–102, Sep. 2000.

[15] D. Zhang and G. Lu, "Generic fourier descriptor for shape-based image retrieval," in *Proc. ICME*, vol. 1. Aug. 2002, pp. 425–428.

[16] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. ICCV*, vol. 2. Sep. 1999, pp. 1150–1157.

[17] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, Jun. 2005, pp. 886–893.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. CVPR*, 2014, pp. 580–587.

[19] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: An astounding baseline for recognition," in *Proc. CVPRW*, 2014, pp. 806–813.

[20] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. CVPR*, 2015, pp. 815–823.

[21] Y. Gao *et al.*, "3D object retrieval with multimodal views," in *Proc. Eurograph. Workshop 3D Object Retr.*, 2015, pp. 129–136.

[22] (2016). *Embedding*. [Online]. Available: https://en.wikipedia.org/wiki/Embedding

[23] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Feb. 2009.

[24] J. Wang *et al.*, "Learning fine-grained image similarity with deep ranking," in *Proc. CVPR*, 2014, pp. 1386–1393.

[25] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proc. ICCV*, 2015, pp. 2794–2802.

[26] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proc. BMVC*, 2015, p. 6.

[27] J. Y.-H. Ng, F. Yang, and L. Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. CVPRW*, 2015, pp. 53–61.

[28] J.-L. Shih, C.-H. Lee, and J. T. Wang, "A new 3D model retrieval approach based on the elevation descriptor," *Pattern Recognit.*, vol. 40, no. 1, pp. 283–295, 2007.

[29] R. Ohbuchi, K. Osada, T. Furuya, and T. Banno, "Salient local visual features for shape-based 3D model retrieval," in *Proc. SMI*, Jan. 2008, pp. 93–102.

[30] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *Proc. CVPR*, vol. 2. Jun. 2005, pp. 524–531.

[31] Z. Lian, A. Godil, and X. Sun, "Visual similarity based 3D shape retrieval using bag-of-features," in *Proc. SMI*, Jun. 2010, pp. 25–36.

[32] P. Papadakis, I. Pratikakis, T. Theoharis, G. Passalis, and S. Perantonis, "3d object retrieval using an efficient and compact hybrid shape descriptor," in *Proc. Eurograph. Workshop 3D Object Retr.*, 2008, pp. 9–16.

[33] K. Simonyan and A. Zisserman. (Sep. 2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: http://arxiv.org/abs/1409.1556

[34] Y. Wang, J. Liu, J. Wang, Y. Li, and H. Lu, "Color names learning using convolutional neural networks," in *Proc. ICIP*, Sep. 2015, pp. 217–221.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.

[36] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. CVPR*, Jun. 2006, pp. 1735–1742.

[37] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. NIPS*, 2014, pp. 1988–1996.

[38] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[39] S. Ding, L. Lin, G. Wang, and H. Chao, "Deep feature learning with relative distance comparison for person re-identification," *Pattern Recognit.*, vol. 48, no. 10, pp. 2993–3003, Oct. 2015.

[40] Y. Jia *et al.*. (Jun. 2014). "Caffe: Convolutional architecture for fast feature embedding." [Online]. Available: https://arxiv.org/abs/1408.5093

[41] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. CVPR*, Jun. 2009, pp. 248–255.

[42] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. ECCV*, 2014, pp. 818–833.

[43] M. J. Atallah, "A linear time algorithm for the hausdorff distance between convex polygons," *Inf. Process. Lett.*, vol. 17, no. 4, pp. 207–209, Nov. 1983.

[44] T. D. Russ, M. W. Koch, and C. Q. Little, "A 2d range hausdorff approach for 3d face recognition," in *Proc. CVPRW*, Sep. 2005, p. 169.

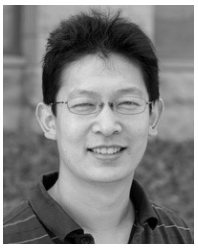
[45] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *Proc. ICRA*, May 2011, pp. 1817–1824.

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

Haiyun Guo received the B.E. degree from Wuhan University, Wuhan, China, in 2013. She is currently pursuing the Ph.D. degree in pattern recognition and intelligence systems with the National Laboratory of Pattern Recognition, Chinese Academy of Sciences. Her current research interests include pattern recognition and machine learning, image and video processing, and intelligent video surveillance.

Jinqiao Wang (M'09) received the B.E. degree from the Hebei University of Technology, China, in 2001, the M.S. degree from Tianjin University, China, in 2004, and the Ph.D. degree in pattern recognition and intelligence systems from the National Laboratory of Pattern Recognition, Chinese Academy of Sciences, in 2008. He is currently an Associate Professor with the Chinese Academy of Sciences. His research interests include pattern recognition and machine learning, image and video processing, mobile multimedia, and intelligent video surveillance.

Yue Gao (SM'14) received the B.E. degree from the Harbin Institute of Technology, Harbin, China, and the M.S. and Ph.D. degrees from Tsinghua University, Beijing, China. He is currently with Key Laboratory for Information System Security, Ministry of Education, Tsinghua National Laboratory for Information Science and Technology (TNList), School of Software, Tsinghua University.



Jianqiang Li received the B.E. degree in mechatronics from the Beijing Institute of Technology, Beijing, China, in 1996, and the M.S. and Ph.D. degrees in control science and engineering from Tsinghua University, Beijing, in 2001 and 2004, respectively. He was a Researcher with the Digital Enterprise Research Institute, National University of Ireland, Galway, from 2004 to 2005. From 2005 to 2013, he was with NEC Labs China as a Researcher, and the Department of Computer Science, Stanford University, as a Visiting Scholar, from 2009 to 2010. He joined the Beijing University of Technology, Beijing, in 2013, as a Beijing Distinguished Professor. He has authored over 40 publications and 37 international patent applications (19 of them have been granted in China, the U.S., or Japan). His research interests are Petri nets, enterprise information systems, business processes, data mining, information retrieval, semantic web, privacy protection, and big data. He served as a PC Member in multiple international conferences and organized the IEEE Workshop on Medical Computing.

Hanqing Lu (SM'06) received the B.E. and M.S. degrees from the Harbin Institute of Technology in 1982 and 1985, respectively, and the Ph.D. degree from the Huazhong University of Sciences and Technology in 1992. He is currently the Deputy Director of the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. His research interests include image and video analysis, medical image processing, and object recognition.