Cross-Domain Image-Based 3D Shape Retrieval by View Sequence Learning

Tang Lee[1], Yen-Liang Lin[2], HungYueh Chiang[1], Ming-Wei Chiu[1], Winston H. Hsu[1], and Polly Huang[1]

[1]National Taiwan University
[2]GE Global Research, Niskayuna, NY, USA

Abstract

We propose a cross-domain image-based 3D shape retrieval method, which learns a joint embedding space for natural images and 3D shapes in an end-to-end manner. The similarities between images and 3D shapes can be computed as the distances in this embedding space. To better encode a 3D shape, we propose a new feature aggregation method, Cross-View Convolution (CVC), which models a 3D shape as a sequence of rendered views. For bridging the gaps between images and 3D shapes, we propose a Cross-Domain Triplet Neural Network (CDTNN) that incorporates an adaptation layer to match the features from different domains better and can be trained end-to-end. In addition, we speed up the triplet training process by presenting a new fast cross-domain triplet neural network architecture. We evaluate our method on a new image to 3D shape dataset for category-level retrieval and ObjectNet3D for instance-level retrieval. Experimental results demonstrate that our method outperforms the state-of-the-art approaches in terms of retrieval performance. We also provide in-depth analysis of various design choices to further reduce the memory storage and computational cost.

1. Introduction

There has been a growing number of applications using 3D shapes. Publicly-available online 3D shape databases (e.g., 3D Warehouse and Thingiverse) enable users to search and download the 3D shapes for various applications (e.g., 3D printing or augmented reality). These databases provide text-based interfaces to let users search 3D shapes by inputting the keywords. However, text-based inputs are often too general (rough) to precisely express the target 3D shapes users intend to retrieve. This recently motivates the image-based 3D shape retrieval (IBSR) research that use images as queries to better retrieve the 3D shapes [12, 7, 14].

Figure 1 shows the idea of our method. A user can input a query photo, and our system will automatically return a list of similar 3D shapes. We propose a new cross-domain learning model to better generate the image and shape representations in a joint embedding space. Therefore, the similarities between images and 3D shapes can be effectively computed by the distances in this space.

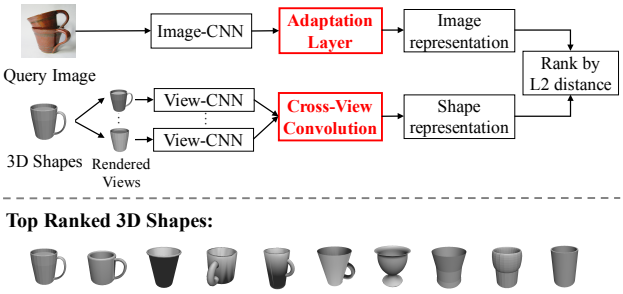

Figure 1. We propose a cross-domain image-based 3D shape retrieval. Given an input query image, our system automatically returns a list of similar 3D shapes by L2 distance. Our method learns a joint image and 3D shape embedding space. Adaptation Layer and Cross-View Convolution are presented to learn better representations for images and 3D shapes respectively. See Figure 2 for more details.

Convolutional neural network (CNN) has been shown powerful on feature representations. As the growth of the number of 3D shapes, various deep learning models are developed for 3D shape recognition [21, 20] and reconstruction [5, 7]. This fact shows a great potential of CNNs to extract meaningful feature representation for 3D shapes. However, there exist two main challenges for cross-domain 3D shape retrieval: (1) how to find a suitable feature representation to better encode a 3D shape, so it can effectively represent a 3D shape and be compared with image features, and (2) how to jointly embed images and 3D shapes into the same feature space.

Most of the prior work mainly focus on 3D shape classification. Two mainstream methods for classifying the 3D shapes are: view-based [21, 20] and voxel-based [25, 1, 16] methods. View-based CNN methods take several rendered

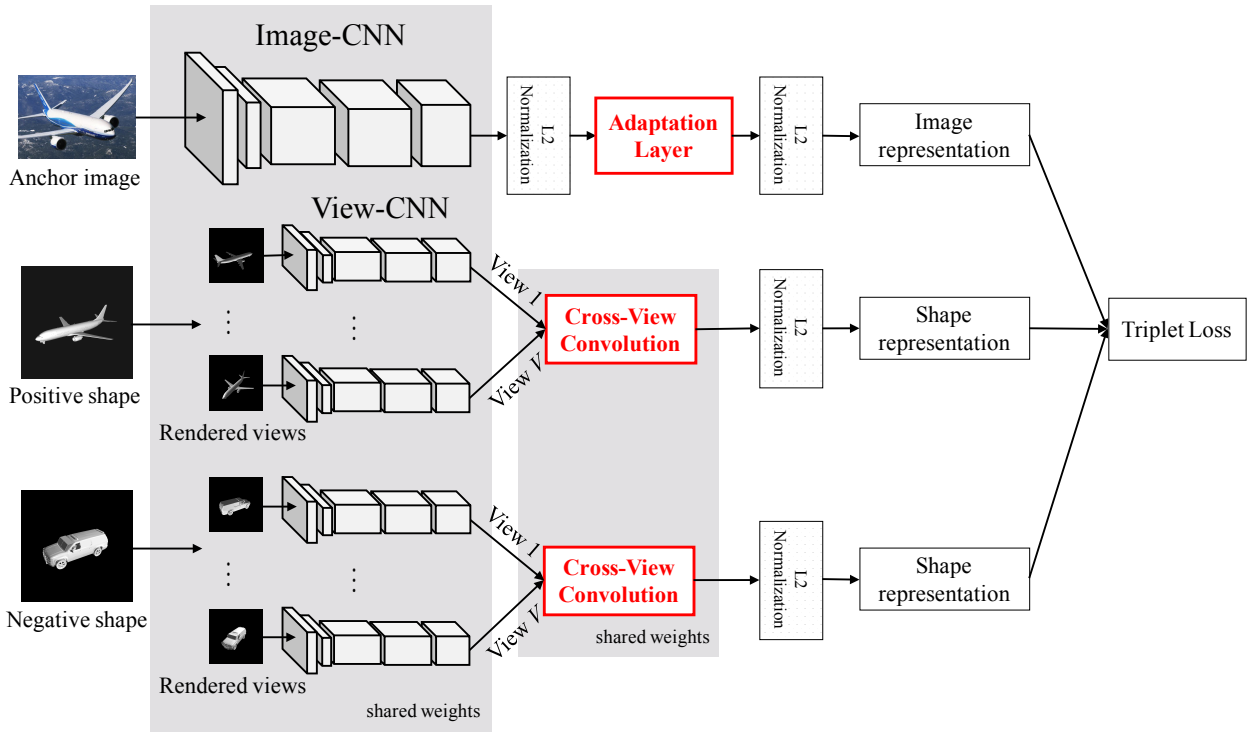

Figure 2. System overview of the proposed end-to-end cross-domain triplet neural network (CDTNN) for image-based 3D shape retrieval. In the training stage, 3D shapes are rendered into a set of consecutive views. Images and rendered views are fed into Image-CNN and View-CNN respectively for feature extraction. To match the features across different domains, we propose an adaptation layer for adapting the real image to the rendered views. For better encoding a 3D shape, we propose a Cross-View Convolution (CVC) layer to aggregate a sequence of views and generate a compact shape representation. Triplet loss is employed to jointly optimize the image and 3D shape representation in an end-to-end manner, which enforces the distance between an anchor image to a negative shape at least farther than the distance between the anchor image to a positive shape by a certain margin.

images as inputs for recognition. They leverage well-exploited 2D CNN models and the large-scale image data [17] for pre-training. In contrast, voxel-based methods use 3D convolution and 3D pooling layers, and process 3D data in the 3D space directly.

Different from prior work for 3D shape classification, our method aims at cross-domain image-based 3D shape retrieval. To better match the features between images and 3D shapes, we adopt view-based CNN methods for encoding a 3D shape. It is crucial to design a view aggregation method to effectively combine the information from different rendered views and generate a compact global feature representation. Instead of using a view-pooling layer to aggregate the features (e.g., max-pooling in MVCNN [21]), we propose a new feature aggregation method, Cross-View Convolution (CVC), which models a 3D shape as a sequence of rendered views that takes sequential information into account. Our experimental results show that our sequence modeling method further improves the results from MVCNN (see Table. 2).

Triplet neural network is commonly used to capture the intra-class and inter-class differences [19, 24, 28]. However, traditional triplet neural networks often adopt the same structure for three streams, which are not directly applicable for our task on cross-domain image-based 3D shape search. To address this problem, we propose a new triplet architecture, Cross-Domain Triplet Neural Network (CDTNN), that incorporates an adaptation layer and can be trained end-to-end. The proposed structure is general and can be adopted for other cross-domain tasks.

We evaluate our method on ObjectNet3D [26] for instance retrieval and our new dataset for category-level retrieval [1]. Our dataset contains 12,311 3D shapes from ModelNet40 [25] and 10,000 images from [17] and Google Image Search. We manually select images that have the objects near the image center (with certain translations and rotations) and include backgrounds around the objects (see examples in Figure 5). The purpose of collecting a new dataset is to better simulate the real situation where an user takes a photo of an object under the unconstrained environment. The main contributions of this paper are summarized as the following:

[1]The dataset and source code are available at: http://cmlab.csie.ntu.edu.tw/~weitang114/ibsr

- We propose an end-to-end cross-domain image-based 3D shape retrieval. It is shown experimentally to outperform state-of-the-art approaches on ObjectNet3D [26] for instance-level retrieval and our new dataset for category-level retrieval.
- We propose a new view aggregation method, Cross-View Convolution (CVC), to effectively encode a 3D shape from a sequence of rendered views, and a cross-domain triplet neural network (CDTNN) that incorporates an adaptation layer to better match the features from different domains.
- We speed up the training of CDTNN by presenting a fast cross-domain triplet neural network architecture, and provide an in-depth analysis of various design decisions to further reduce the memory storage and computational cost.

2. Related Work

3D Shape Classification: various methods have been proposed for 3D shape classification: view-based [21, 20] and voxel-based [25, 1, 16]. View-based models leverage the well-exploited 2D CNN models and large-scale image data [17] for pre-training. Su et al. [21] (MVCNN) project a 3D shape into a fixed number of views. Each view is fed into a CNN to extract features. A view-pooling layer is used to aggregate the features from different views and generate a global feature representation. Finally, the view-pooled feature representation is fed into fully connected layers for classification. However, MVCNN uses a max-pooling layer to aggregate features, which ignores the sequential information between consecutive views. To better leverage the sequential information, we propose a new view aggregation method that takes the view sequences into account and shows better results than view-pooling.

3D Shape Retrieval: Wang et al. [23] propose a sketch-based 3D shape retrieval by Siamese Networks. They train two Siamese Networks to extract features from sketches and two rendered views of a 3D shape. Instead of using two rendered views, our model is capable of utilizing a sequence of views of a 3D shape at the same time. Massa et al. [14] leverage 3D CAD models for object detection in 2D images. To match the features from two different domains, they use an adaptation layer for adapting real images to rendered views. Our adaptation layer is different from theirs in two folds. First, they do not train their model in an end-to-end manner, i.e., they keep CNN layers fixed and only train the adaptation layer individually. We train our cross-domain triplet network in an end-to-end manner. Second, they only use synthetic images for training the adaptation layer, which still has an appearance gap for real images. Instead, we train our model with real images to further reduce the gaps between two different domains.

Li *et al*. [12] propose a framework to jointly embed images and 3D shapes into the same space. The embedding space is built based on the distances of HOG features from 3D shapes. However, their method is not scalable for large-scale 3D shape datasets, as it requires to compute the distances for all training 3D shape pairs. Also, their method is not trained end-to-end. Tasse *et al*. [22] build a multi-modal 3D shape retrieval system by embedding images, sketches and 3D shapes in the same vector space based on word vector. Their method assumes that during training, the category labels are given, while our method uses only the pairwise similarities for training.

Different from prior methods for 3D shape classification, our method aims at cross-domain image-based 3D shape retrieval. We propose an end-to-end Cross-Domain Triplet Neural Network (CDTNN) for matching the features from two different domains. To better encode a 3D shape, we propose a Cross-View Convolution (CVC) to aggregate a sequence of views. We also improve the training speed and memory storage with a fast cross-domain triplet neural network architecture.

3. Method

3.1. 3D Shape Representation

To match the 3D shapes with the 2D images, we render V views of each 3D shape by Phong Shading [15] with a black background. We propose a cross-domain network to bridge the gaps between the real images and non-photorealistic rendered 3D views. Each rendered view is fed into a View-CNN, and the feature maps are aggregated to generate a global representation of a 3D shape. We adopt the AlexNet [9] for Image-CNN and View-CNN. The parameters from different View-CNNs are shared. The pool5 feature map is selected as the feature representation.

We use AlexNet for fairly comparing with MVCNN [21]. AlexNet has a similar model complexity with their base model VGG-M [4]. We believe that using more sophisticated models can further improve the performance, but the investigation of different base models is out of the scope of this work.

3.2. Cross-View Convolution (CVC)

For view-pooling in MVCNN [21], the feature maps of different views are max-pooled and generate a global feature for representing a 3D shape. However, max-pooling is unaware of the sequential information of the consecutive views and treats each view independently. The motivation behind our method is that the rendering process for V views can be seen as sequentially taking photos around an object. Each rendered view is correlated to its consecutive views. Furthermore, some views are more important than the others. For example, the back view of a "dresser" looks very

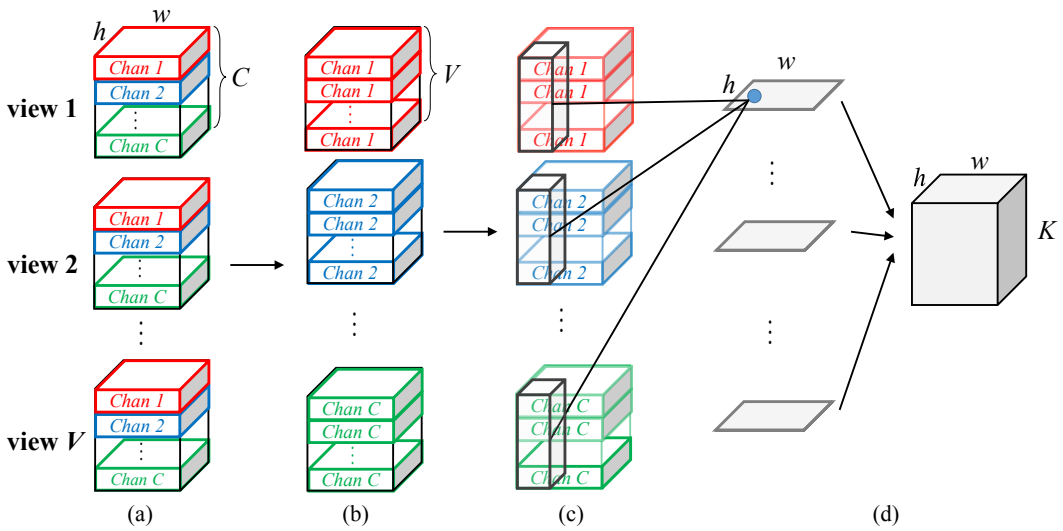

Figure 3. Cross-View Convolution (CVC) is proposed to aggregate the features from the consecutive views into a compact 3D shape representation. (a) The feature map for each of the V views is a $w \times h \times C$ tensor, where w is width, h is height, and C is the number of channels. (b) The tensors from different views are stacked together and generate $C \times (w \times h \times V)$ tensors, which are convolved by K CVC 3D kernels (c). The size of each 3D kernel is $1 \times 1 \times V \times C$. K feature maps are generated (d) and stacked together to obtain a $w \times h \times K$ feature map (e). The feature map is then flattened to a vector for 3D shape representation. (Best viewed in color)

similar to the back view of a "bookshelf," i.e., we should give lower weights to the back view for a dresser. We assume that the 3D shapes are upright oriented along a consistent axis as in [25, 23], since most of 3D CAD models in the current online databases, e.g., ShapeNet [3] and 3D Warehouse, are human-designed and often have common orientations.

We propose a Cross-View Convolution (CVC) layer to better exploit the sequential information, which is not well exploited in the prior literatures. Specifically, the size of the feature map for each of the V views is $w \times h \times C$, where w is width, h is height and C is the number of channels (Figure 3 (a)). We stack the tensors of each channel from different views and generate $C \times (w \times h \times V)$ tensors (Figure 3 (b)). We convolve the tensors with K CVC kernels (Figure 3 (c)) and generate K feature maps (Figure 3 (d)). Finally, K feature maps are stacked together for the shape representation (Figure 3 (e)). The CVC kernel can be seen as a weighted summation across different views to model the sequential information. To make the output channel number consistent with the input, we set $K = C = 256$ in our experiments.

3.3. Jointly Embedding Images and 3D Shapes

The CNNs for images (Image-CNN) and views (View-CNN) are learned jointly to construct a joint embedding space. We use triplet neural network architecture and propose a fast triplet architecture to speed up the training. The goal of triplet neural network is to enforce the anchor-negative distances at least farther than the anchor-positive distances by a certain margin:

$$d_{pos} - d_{neg} + margin < 0, \tag{1}$$

where d_{pos} is the anchor-positive distance and d_{neg} is the anchor-negative distance. The three streams in the triplet network are anchor image, positive shape, and negative shape (see Figure 2). We use Image-CNN with the adaption layer for the anchor image stream and View-CNNs with the CVC layer for the positive and negative shape streams. The triplet loss is defined as:

$$Loss_{triplet} = d_{pos} - d_{neg} + margin \tag{2}$$

We only consider the triplets that break the constraints and the final loss becomes:

$$Loss = \begin{cases} Loss_{triplet} & \text{if } Loss_{triplet} > 0 \\ 0 & \text{Otherwise.} \end{cases} \tag{3}$$

The features are normalized by their L2 norm. The distance metric is the squared Euclidean distance. Image-CNN and V streams of View-CNN are shared with the same parameters. Since Image-CNN and View-CNN stem from two different domains, directly using the same parameters will lead to the performance drop. In the next section, we will explain how to share the parameters of CNNs from two different domains without deteriorating the performance.

3.3.1 Adaptation Layer

To bridge the gaps between image and 3D shape feature representations, we propose to utilize an adaptation layer after

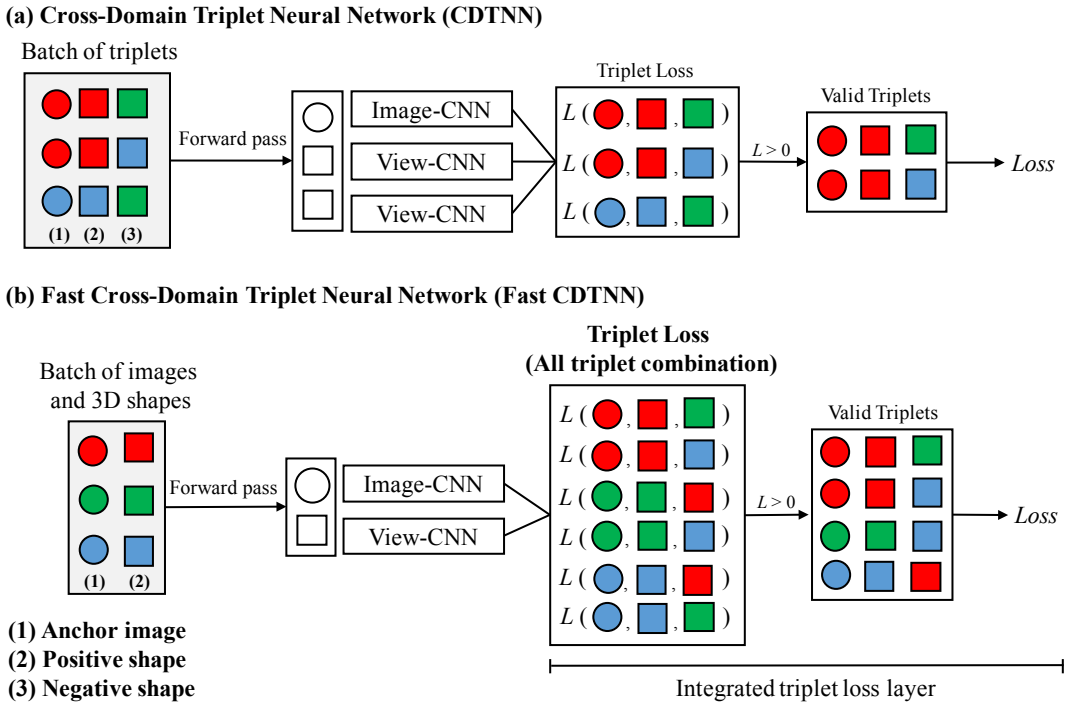

Figure 4. The architecture of CDTNN and Fast-CDTNN. The circles and squares represent the anchor images and 3D shapes respectively, and colors denote different categories. Instead of sampling the triplets before the forward passes, we enumerate the triplets after the forward passes, which generates more triplets while requiring the similar (or less) computing cost for each batch. As a result, Fast-CDTNN converges much faster than CDTNN. (Best viewed in color)

Image-CNN. We choose to add the adaptation layer after Image-CNN rather than View-CNN, since it can be utilized as attention masks to detect the salient objects in the images (the query images of our dataset are not always centralized). The proposed Cross-Domain Triplet Neural Network (CDTNN) with the adaptation layer jointly learns a triplet network for different domains end-to-end. We use a fully connected layer followed by an ReLU layer for the adaptation layer. The output size of the fully connected layer is equal to the input size of the feature map.

3.3.2 Fast Cross-Domain Triplet Neural Network

We propose a new Fast-CDTNN architecture to further speed up the training of CDTNN. Our idea is that instead of sampling the triplets before the forward passes in a similar way as prior triplet-based methods [24, 19, 8] (see Figure 4 (a)), we enumerate all possible triplets after the forward passes (see Figure 4 (b)). Different from [2] for a single-domain triplet neural network, our triplet neural network involves cross-domain triplets. The advantages of the triple enumeration method are: First, the CNN features of the same images and 3D shapes do not need to be re-computed. Second, each image or shape can contribute to many triplet losses.

	CDTNN	Fast-CDTNN
# Images	P	P
# Shapes	$2P$	QC
# Triplets	P	$\boldsymbol{PQ^2(C-1)}$
# Forward passes	$P + 2P \times V$	$P + QC \times V$

Table 1. Comparison of the computational cost and triplet numbers in a batch for CDTNN and Fast-CDTNN. Q is the number of shapes for each category, C is the number of classes and V is the number of views. We multiply V (views) for computing the number of forward passes for 3D shapes, as each rendered view is fed into a View-CNN. Fast-CDTNN requires similar (or less) forward numbers (i.e., passing through Image-CNN and View-CNN), but can generate much more triplets than CDTNN. For the examples shown in Figure 4, we have $P = 3$, $Q = 1$, $C = 3$ and $V = 12$. CDTNN generates 3 triplets and requires 75 forward passes, while Fast-CDTNN generates 6 triplets and only requires 39 forward passes.

Thus, our Fast-CDTNN generates more triplets while requiring similar computational cost as CDTNN (see Table 1). As a result, Fast-CDTNN converges faster than CDTNN, i.e., approximately 5$\times$ faster in the training time of our experiments.

We denote the triplet loss L_{ijk} for the i-th image, j-th positive 3D shape and k-th negative 3D shape. Each triplet

loss is computed by Eq. 2. We filter out the triplets whose values are less than zero and only keep the positive ones [19]. The final loss is the mean of all valid triplet losses:

$$Loss = \frac{1}{N} \sum_{i=1}^{P} \sum_{j=1}^{Q} \sum_{k=1}^{Q(C-1)} ReLU(L_{ijk}), \qquad (4)$$

where P is the number of images, Q is the number of shapes for each category, C is the number of classes and N is the number of valid triplet losses. Each image has Q positive shapes and $Q(C-1)$ negative shapes (we use the shapes from other categories as negatives), so we generate total $PQ^2(C-1)$ triplets for each batch. We set $\mathrm{ReLU}'(0) = 0$ to handle the non-differentiability of the ReLU function for computing the back-propagation.

Moreover, we integrate the triplet combination, triplet filtering, and loss into a single triplet loss layer. The loss computation can be done in a single forward-backward pass to further reduce the computing time. We set $P = 50, Q = 3, V = 12$, and $C = 40$ for our experiments. For each batch, Fast-CDTNN generates 17,550 triplets, which are significantly more than the 50 triplets used in original CDTNN.

3.4. Implementation

We implement our models by TensorFlow [6]. The parameters of the layers in Image-CNN and View-CNN are pre-trained by ImageNet 2012 [17]. The weights of Cross-View Convolution and Adaptation Layer are initialized with Orthogonal Initialization [18] and their biases are initialized to 0. We set $margin = 0.2$ in triplet loss and use Adam Optimizer for training the neural networks. We conduct our experiments on a Linux workstation with an Intel Xeon E5-2650 v3 CPU, 128 GB memory, and one NVIDIA Tesla K80 GPU. It takes approximately 30 hours for our fast cross-domain triplet neural network to converge.

3.5. Data Preprocessing

We process the images for the data augmentation in the training phase. The original size of images is 256×256. We randomly crop a patch of 227×227 and flip it horizontally. We subtract from the images the mean values of the RGB channels from ImageNet 2012 [17]. Then, we convert the images from RGB into grayscale to make our network learn more shape information and insensitive to the color variations. We render $V = 12$ views for each 3D shape using Phong Shading [15] by placing 12 virtual cameras around the 3D shape every 30 degrees. The cameras are elevated 30 degrees from the ground plane.

4. Experiments

4.1. Dataset

Existing image to 3D shape datasets [13, 27] are not suitable for our purposes since the size of these datasets are too small to train our network. We evaluate our method on ObjectNet3D [26] for instance-level object retrieval, which contains 100 categories and 44,147 3D shapes. To better simulate the real situation where an user takes photos under unconstrained environment, we build a new large-scale image to 3D shape dataset. The dataset consists of 12,311 3D shapes, 10,000 images, and 40 categories. Images and 3D shapes are annotated with category labels. There are totally 9,843 3D shapes and 8,000 images for training, and 2,468 3D shapes and 2,000 images for testing. The 3D shapes and images are adopted from ModelNet40 [25] and ImageNet 2012 [17] respectively. For three missing categories in ImageNet: xbox, glass box, and tv stand, we crawl 250 images for each of them from Google Image Search. There are totally 250 images for each category. 200 images are for training and 50 images are for testing.

4.2. Image-based 3D Shape Retrieval

We conduct the experiments for image-based 3D shape retrieval on our dataset. A 3D shape is considered relevant to a query image if they belong to the same category, following the standard evaluation criteria for cross-domain 3D shape retrieval [23, 10, 11, 22]. Note that our method only requires similarities for training, and class label is only used to obtain the similarity. We extract the features for both query images and 3D shapes with our CDTNN network and compute the L2 distances for ranking the 3D shapes. The retrieval performance is evaluated by mean average precision (mAP).

Table 2 shows the mAP for different methods. We compare our method with three baseline methods: AlexNet [9], MVCNN [21], and an augmented MVCNN with triplet network. For AlexNet, we extract the features of the query images and rendered views at the pool5 layer of a pre-trained AlexNet model. The distance is computed as the minimum distance between the query image and 12 rendered views of a 3D shape. For MVCNN, we follow the original method in [21] for cross-domain 3D shape retrieval (they use it for sketch-based 3D shape retrieval, while we adopt it for image-based 3D shape retrieval). Image features are extracted from the pool5 layer of a pre-trained AlexNet model, and 3D shape features are extracted from the view-pooling layer of MVCNN. The MVCNN is based on a pre-trained AlexNet (we tried with a fine-tuned AlexNet, but it gets even worse results due to the large domain differences). The distances are computed as the L2 distances between the normalized image features and 3D shape features. These experiments show that there exists an large domain gap be-

Method	mAP
AlexNet pool5	7.16%
MVCNN [21]	7.92%
Triplet + MVCNN	40.85%
CDTNN + Adaptation Layer	47.84%
CDTNN + Adaptation Layer + CVC	**52.67%**

Table 2. Retrieval performance (mAP) for cross-domain image-based 3D shape retrieval.

tween images and 3D shape views that can not be bridged with simple CNN features.

We augment the original MVCNN with triplet network as a strong baseline approach (Triplet MVCNN in Table 2). We use AlexNet for the anchor image stream and MVCNN with view-pooling for the positive and negative streams in the triplet network. Comparing to the original MVCNN for cross-domain retrieval [21], our triplet MVCNN explicitly learns the cross domain image-shape pairs and improves the mAP to 40.85%.

Our CDTNN model (CDTNN + Adaptation Layer) with adaptation layer further improves the triplet MVCNN to 47.84%. It shows the effectiveness of the adaptation layer for bridging the gaps between two different domains. Finally, incorporating the CVC layer (CDTNN + Adaptation Layer + CVC) achieves the best performance 52.67%. The experimental results confirm that end-to-end Cross-Domain Triplet Neural Network (CDTNN), Adaptation Layer and Cross-View Convolution (CVC) are very effective for image-based 3D shape retrieval. Some results are visualized in Figure 5.

4.3. Ablation Study

We are interested in whether the adaptation layer is necessary for the cross-domain learning. To clarify this, we investigate four different settings, sharing or not sharing the weights (i.e., sharing the weights for Image-CNN and View-CNN) and whether to include an adaptation layer. These experiments are conducted with CDTNN with CVC layers.

Table 3 shows the retrieval results. The shared-weight model with the adaptation layer shows the best performance. The results reveal that the adaptation layer improves the results for both shared and unshared weights. In addition, it can significantly reduce the model parameters by sharing the weights between Image-CNN and View-CNN without deteriorating the retrieval performance.

4.4. Instance-level IBSR

Our CDTNN only requires the pair-wise similarities during training, it can be easily adopted into instance-level retrieval, i.e., images and 3D shapes belong to the same instance are "similar" pairs and others are "dissimilar" pairs. The goal of instance-level retrieval is to retrieve the exact 3D shape instances to the query objects. We compare our

Shared weights	Adaptation layer	mAP
No	No	45.49%
No	Yes	52.02%
Yes	No	44.49%
Yes	Yes	52.67%

Table 3. Adaptation layer is very effective for learning the cross-domain triplet neural network (see Figure 2). Also, it can significantly reduce the model parameters (i.e., by sharing the weights between View-CNN and Image-CNN) without any performance drop.

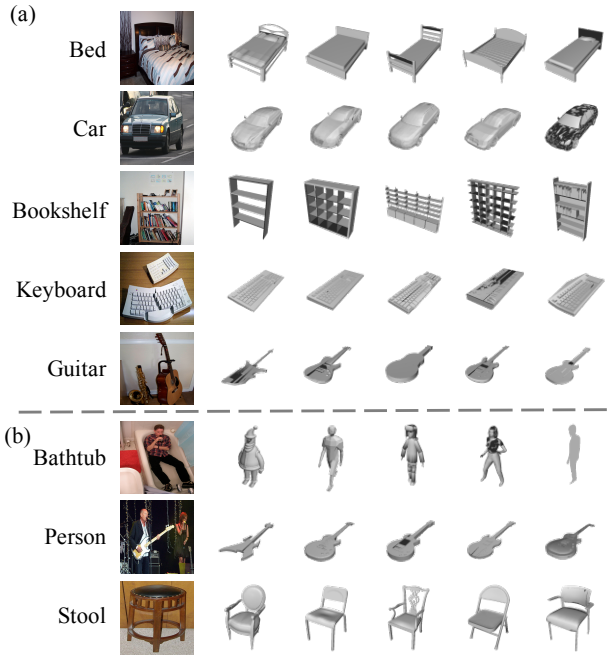

Figure 5. Example retrieval results of our system. (a) Successfully retrieved 3D shapes. Our method retrieves similar 3D shapes with 2D query images, even if the objects are not in the central positions and presented with various scales. (b) Some failure cases of our system are mainly caused by severe occlusion (e.g., a person lies on a bathtub), multiple objects (e.g., a person holds a guitar) and similar objects (stool vs. chairs), showing that our dataset is very challenging.

method with the state-of-the-art method: Lifted-Structured Embedding (LSE) [26], and evaluate the performance on three categories of ObjectNet3D: aeroplane, car, and chair. We quantitatively evaluate the retrieval performance by using the annotations in ObjectNet3D.

For each category, we split the 80% of training images for training and 20% for testing, as ObjectNet3D does not have annotations for the testing images. We crop each object in the images using the bounding box annotations. Each object is used as a query image for training and testing. We use Recall@K for our evaluation metric. Table 4 shows the results. Original LSE [26] only uses the synthetic images for training the embedding space. For fairly comparing to

Category	aeroplane (384 shapes)			car (1630 shapes)			chair (1387 shapes)		
Method	LSE	LSE (+real)	**CDTNN**	LSE	LSE (+real)	**CDTNN**	LSE	LSE (+real)	**CDTNN**
R@1	0.028	0.009	**0.042**	0.000	0.017	**0.029**	0.006	0.015	**0.067**
R@10	0.079	0.102	**0.206**	0.004	0.069	**0.182**	0.038	0.106	**0.227**
R@20	0.116	0.181	**0.294**	0.020	0.103	**0.270**	0.073	0.160	**0.329**
R@50	0.208	0.319	**0.483**	0.044	0.166	**0.450**	0.141	0.272	**0.485**
R@100	0.421	0.495	**0.660**	0.082	0.240	**0.610**	0.221	0.398	**0.617**

Table 4. Results of instance-level image-based 3D shape retrieval. Our method outperforms LSE [26] in all categories. For fair comparison, we augment their method by using the real images for training (LSE (+real)). The evaluation metric is Recall@K (the higher the better).

our method, we augment it by incorporating the real images in the training (LSE (+real)). The results demonstrate that CDTNN significantly outperforms LSE and LSE (+real) in all categories.

5. Conclusions

In this work, we propose a Cross-Domain Triplet Neural Network (CDTNN) for image-based 3D shape retrieval. We introduce a new Cross-View Convolution (CVC) layer to better model the view sequences of a 3D shape. For jointly training the cross-domain model in an end-to-end manner, we investigate an adaptation layer for converting the real images to the rendered views, and automatically extracting the salient objects in the images. We speed up the triplet training by presenting a new fast CDTNN architecture. Experimental results validate the effectiveness of the proposed methods on both category-level and instance-level retrieval. For future work, we seek further improvements by incorporating the region-based methods (e.g., object proposal) for handling multiple objects in an image, and extracting salient parts for improving the fine-grained 3D shape retrieval.

Acknowledgement

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant 105-2218-E-002-011 and 107-2634-F-002-007. We also benefit from the grants from NVIDIA and the NVIDIA DGX-1 AI Supercomputer.

References

[1] M. Allen, L. Girod, R. Newton, S. Madden, D. T. Blumstein, and D. Estrin. Voxnet: An interactive, rapidly-deployable acoustic monitoring platform. In *IPSN*, 2008. 1, 3

[2] B. Amos, B. Ludwiczuk, and M. Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016. 5

[3] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 4

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014. 3

[5] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016. 1

[6] M. A. et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 2015. 6

[7] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 1

[8] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, 2015. 5

[9] A. Krizhevsky, I. Sutskever, and G. E. H. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3, 6

[10] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, Q. Chen, N. K. Chowdhury, B. Fang, et al. A comparison of 3d shape retrieval methods based on a large-scale benchmark supporting multimodal queries. *Computer Vision and Image Understanding*, 131:1–27, 2015. 6

[11] B. Li, Y. Lu, C. Li, A. Godil, T. Schreck, M. Aono, M. Burtscher, H. Fu, T. Furuya, H. Johan, et al. Shrec'14 track: extended large scale sketch-based 3d shape retrieval. In *Eurographics workshop on 3D object retrieval*, volume 2014, 2014. 6

[12] 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 Transactions on Graph*, 2015. 1, 3

[13] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *ICCV*, 2013. 6

[14] F. Massa, B. Russell, and M. Aubry. Deep exemplar 2d-3d detection by adapting from real to rendered views. In *CVPR*, 2016. 1, 3

[15] B. T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 1975. 3, 6

[16] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5648–5656, 2016. 1, 3

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein,

A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 2, 3, 6

[18] A. M. Saxe, J. L. McClelland, and S. Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *arXiv preprint arXiv:1312.6120*, 2013. 6

[19] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 2, 5, 6

[20] B. Shi, S. Bai, Z. Zhou, and X. Bai. Deeppano: Deep panoramic representation for 3d shape recognition. *IEEE Signal Processing Letters*, 2015. 1, 3

[21] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015. 1, 2, 3, 6, 7

[22] F. P. Tasse and N. Dodgson. Shape2vec: semantic-based descriptors for 3d shapes, sketches and images. *TOG*, 2016. 3, 6

[23] F. Wang, L. Kang, and Y. Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *CVPR*, 2015. 3, 4, 6

[24] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014. 2, 5

[25] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015. 1, 2, 3, 4, 6

[26] Y. Xiang, W. Kim, W. Chen, J. Ji, C. Choy, H. Su, R. Mottaghi, L. Guibas, and S. Savarese. Objectnet3d: A large scale database for 3d object recognition. In *ECCV*, 2016. 2, 3, 6, 7, 8

[27] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, 2014. 6

[28] Q. Yu, F. Liu, Y.-Z. Song, T. Xiang, T. M. Hospedales, and C. C. Loy. Sketch me that shoe. In *CVPR*, 2016. 2