Triplet-Center Loss for Multi-View 3D Object Retrieval

Xinwei He Yang Zhou* Zhichao Zhou Song Bai Xiang Bai
Huazhong University of Science and Technology
{eriche.hust, zhouyangvcc}@gmail.com {zhichaozhou,songbai,xbai}@hust.edu.cn

Abstract

Most existing 3D object recognition algorithms focus on leveraging the strong discriminative power of deep learning models with softmax loss for the classification of 3D data, while learning discriminative features with deep metric learning for 3D object retrieval is more or less neglected.

In the paper, we study variants of deep metric learning losses for 3D object retrieval, which did not receive enough attention from this area. First , two kinds of representative losses, triplet loss and center loss, are introduced which could learn more discriminative features than traditional classification loss. Then, we propose a novel loss named triplet-center loss, which can further enhance the discriminative power of the features. The proposed triplet-center loss learns a center for each class and requires that the distances between samples and centers from the same class are closer than those from different classes. Extensive experimental results on two popular 3D object retrieval benchmarks and two widely-adopted sketch-based 3D shape retrieval benchmarks consistently demonstrate the effectiveness of our proposed loss, and significant improvements have been achieved compared with the state-of-the-arts.

1. Introduction

In the past few years, 3D shape analysis has received extensive attention from both computer vision and graphics communities. Especially, many new attempts have been made to this field, thanks to the powerful deep learning approaches and the large scale 3D model benchmarks such as ShapeNet [7]. 3D object retrieval is a fundamental issue in shape analysis that is the most crucial for processing and analyzing 3D data. However, most deep learning based approaches focus on leveraging the strong discriminative power of deep learning models for the classification of 3D data, *e.g.*, [33, 14, 25], only a few novel deep learning based approaches specifically designed for 3D object retrieval in large scale have been presented.

With a long history in the community, 3D object retrieval may be coarsely divided into two categories: *view-based* and *model-based* methods. View-based methods [1, 33] extract or learn the shape features from a set of 2D view projections, where 2D convolutional neural networks (CNN) are often adopted to process such projection images. Model-based methods [11, 38] obtain the 3D shape features directly from the original 3D representations so that 3D CNN is preferred. Until now, view-based methods usually outperform model-based ones in term of retrieval accuracy, as reported in the recent competitions of large scale 3D SHape REtrieval Contest (SHREC) [4, 27, 28].

One well-known example of 3D object retrieval is Multi-View Convolutional Neural Networks (MVCNN) [33], a combination of multiple 2D projection features learned by CNN within an end-to-end trainable fashion. Similar to MVCNN, great efforts have been made to build a unified deep learning model that can simultaneously perform the tasks of 3D object classification and retrieval. These approaches including MVCNN believe that a strong classification model trained with deep learning, often can meanwhile provide a faithful similarity for 3D object retrieval.

In fact, deep learning approaches for 3D object retrieval are quite similar to those for image or other object retrieval, where several loss functions such as contrastive loss [8] and triplet loss [29] have been introduced for training CNN, in order to learn a metric or an embedding space that makes the instances from the same category closer to each other than those from different categories. In particular, training a plain CNN model with the triplet loss for end-to-end metric learning has shown its advantages in face recognition [29] and person re-identification (re-ID) [12]. Though the remarkable progresses in the tasks of re-ID in 2D image sets have been achieved using such loss functions, they are more or less neglected by the area of 3D object retrieval. Indeed most existing deep learning approaches for 3D shape retrieval focus on designing the sophisticated architectures of deep neural networks or exploiting different representations of 3D object.

In contrast to most existing algorithms, in this paper we

*Corresponding author.

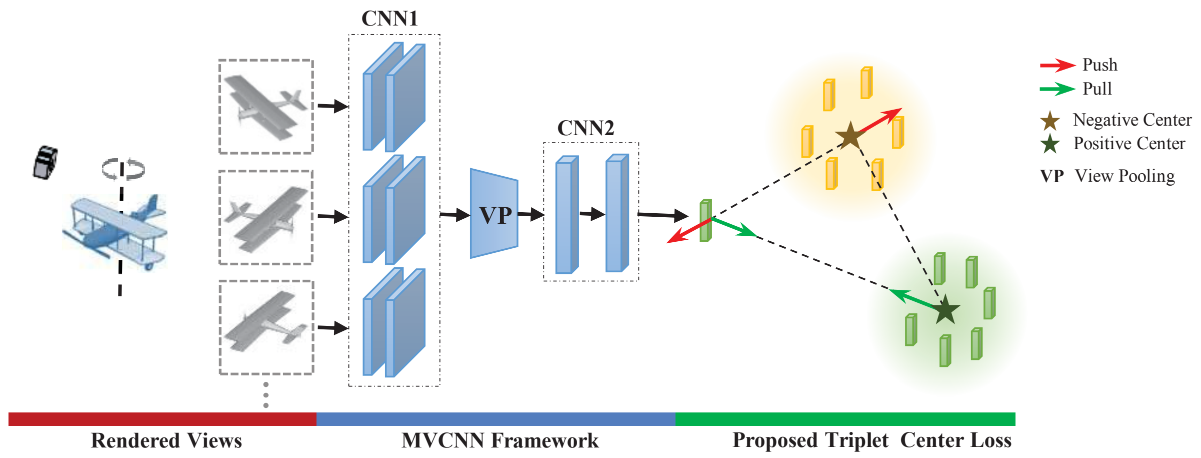

Figure 1. An overview of architecture for 3D object retrieval. We adopt MVCNN as the basic component for achieving view-based 3D object representations, and the proposed TCL is used as the supervision loss. In addition, softmax loss could be also combined into the framework for boosting performance.

argue that training a CNN with the triplet loss [29] or center loss [39] that is specific for distance measure, can also bring the performance benefits to 3D object retrieval, significantly outperforming the state-of-the-art approaches on the most popular benchmark datasets of 3D object retrieval, such as ModelNet40 and ShapeNet Core55.

In summary, we make the following contributions: 1) We firstly introduce two kinds of typical loss functions that are suggested for 3D object retrieval, and fully investigate their impact on the retrieval performance; 2) We propose a novel loss function named triplet-center loss (TCL), and show that the state-of-the-art results are obtained when using TCL based on the same CNN model, superior to other alternatives.

The proposed TCL, motivated by center loss [39] and triplet loss [29], learns a center for each class and requires that the distances between samples and centers from the same class are smaller than those from different classes, such that the features of samples from the same class are pulled closer to the corresponding center and meanwhile pushed away from the other centers of different classes. Different from center loss which only focuses on reducing the intra-class variations, TCL also considers inter-class separability. Compared with triplet loss, TCL avoids the complex construction of triplets and hard sample mining mechanism. With TCL, our CNN model for 3D object retrieval is built upon the framework of MVCNN [33], as illustrated in Figure 1. Therefore, our method can be considered as a view-based approach that unifies the extraction of 3D shape features and distance metric learning into an end-to-end learning procedure. Beyond ModelNet40 and ShapeNet Core55, we also demonstrate its advantages in sketch-based 3D shape retrieval task on two widely-adopted benchmarks, SHREC'13 and SHREC'14 sketch track benchmark datasets respectively.

2. Related work

With the availability of large-scale labeled 3D shape collections like ShapeNet [7], a growing body of literature on 3D shape analysis especially on deep learning has emerged recently. We refer the readers to [34, 13] for a comprehensive survey of 3D shape retrieval. In this section, we will mainly focus on representative 3D shape retrieval methods based on deep learning mechanisms.

In general, 3D shape retrieval methods could be roughly categorized into two classes: 3D model-based methods and view-based methods. 3D model-based methods directly learn shape features from 3D data formats, such as polygon meshes or surfaces [5, 6, 42, 43], voxel grid [21, 40, 18, 25, 30], and point clouds [24, 26]. For example, Furuya *et al.* [11] propose DLAN network to process local regions of 3D shape directly and aggregate local 3D rotation-invariant features to perform retrieval task. Klokov *et al.* [15] propose Kd-network to work with unstructured point clouds and use the learned features to perform retrieval task. The main limitations of these methods lie in the restriction of shape representation (*e.g.*, smooth manifold), or high computational complexity, especially for the voxel-based methods. More recently, Wang *et al.* [38] propose a 3D CNN based on octree representation, which can largely improve the computation efficiency compared with traditional full-voxel-based representations.

The view-based methods usually render a single view or multiple views for a 3D shape firstly, such that sophisticated image feature extractors like CNN can be exploited to extract features from the 2D rendered view, then these extracted view features are assembled into a compact shape descriptor which is finally employed for the retrieval or classification task. For example, MVCNN [33] utilizes a max-pooling layer to aggregate the features of different views

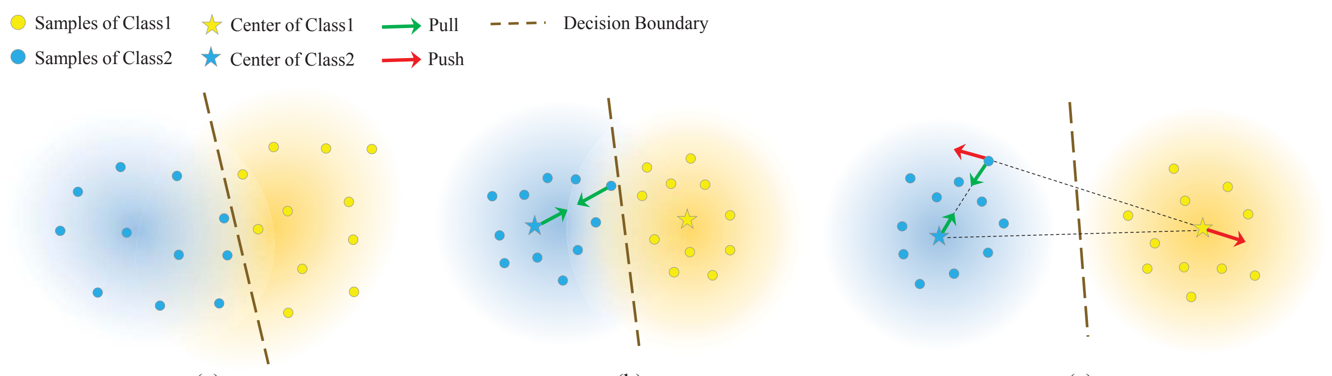

Figure 2. A toy illustration of the distributions of deep features learned by (a) softmax loss, (b) center loss + softmax loss, and (c) triplet-center loss + softmax loss. Intuitively, the decision boundary of the softmax classifier separates the two classes elaborately. The center loss pulls features toward their corresponding centers. The TCL pulls the features to their corresponding centers and pushes the features away from the other centers.

extracted by a shared CNN. Bai *et al.* [1, 2] propose a deep learning-based 3D shape search engine named GIFT, which particularly focuses on the real-time property and scalability of shape retrieval. To this end, GPU and inverted file are used to accelerate the view feature extraction and indexing, and excellent retrieval performances on various shape benchmarks are achieved. Typically, the learned features from view-based methods are more discriminative for 3D shapes, leading to a better retrieval performance in most cases.

However, most of the above methods are not specifically designed for the 3D shape retrieval task, while in image retrieval community, in order to learn more robust and discriminative features, deep metric learning has been widely adopted. Triplet loss [29], which is proposed by Weinberger and Saul, encourages features of data points with the same identity to get closer than those with different identities. Several variants of triplet loss have also been proposed, such as [19, 23, 37]. However, triplet loss may suffer from the problem of time-consuming mining of hard triplets and dramatic data expansion. Aimed at solving this problem, recently Hermans *et al.* [12] propose a batch-hard based triplet loss (BHL) which mines hard negative and hard positive samples from the on-line training batches, and achieve state-of-the-art results on several person re-ID benchmarks. On the other hand, center loss [39] has been brought up which serves as an auxiliary loss for softmax loss for the sake of learning more discriminative features for the problem of face verification. The main objective of center loss is to learn a center for the features of each class and pull features of the same class to the corresponding center more closely.

Inspired by the success of the applications of deep metric learning approaches for 2D image retrieval/re-ID tasks, we introduce two kinds of representative deep metric learning loss, *i.e.*, triplet loss and center loss, to 3D object retrieval. In addition, a novel loss named triplet-center loss is put forward. Recently, Wang *et al.* [36] propose a similar loss for face verification problem. However, our triplet-center loss comes from a very different intuition[1]. Besides, our loss eliminates the need for normalization on features and weights and does not reuse weights from fully connected layer. Remarkable improvements over the state-of-the-art on two 3D shape retrieval benchmarks and two 3D sketch-based retrieval benchmarks demonstrate its effectiveness.

3. Proposed method

In the shape retrieval task, obtaining a robust and discriminative representation of a shape is crucial for obtaining good performance. Usually, this can be partly achieved by exploiting softmax loss to train a CNN on the labeled training set. However, the learned features optimized with the supervision of softmax loss are not discriminative enough in nature, since they only focus on finding a decision boundary to separate shapes of different classes, without considering the intra-class compactness of the features. As illustrated in Fig. 2 (a), although samples of the two classes are separated by the decision boundary elaborately, there exists significant intra-class variations. To cope with this problem, a lot of deep metric learning algorithms have been put forward. Here we first introduce two kinds of representative losses, *i.e.*, triplet loss [29] and center loss [39]. Then, based on these two losses, we derive our proposed TCL.

3.1. Review on triplet loss

Triplet loss, as its name suggested, is calculated on the triplet of training samples (x_a^i, x_+^i, x_-^i), where (x_+^i, x_a^i) have the same class labels and (x_-^i, x_a^i) have different class labels. x_a^i is usually taken as an anchor of a triplet. Intuitively, triplet loss encourages to find an embedding space

[1] [36] is motivated by normalization on weights and features, while our inspiration comes from triplet loss and center loss.

where the distances between samples from the same classes (*i.e.*, x^i_+ and x^i_a) are smaller than those from different classes (*i.e.*, x^i_- and x^i_a) by at least a margin m. Specifically, the triplet loss could be computed as follows:

$$L_{tpl} = \sum_{i=1}^{N} \max\Big(0, m + D\big(f(x^i_a), f(x^i_+)\big) \\ - D\big(f(x^i_a), f(x^i_-)\big)\Big) \tag{1}$$

where $f(\cdot)$ represents the feature embedding output from the neural networks, $D(\cdot)$ measures the distance between two input vectors, N stands for the number of triplets in the training set and i denotes the i-th triplet. However, the number of triplets grows cubically when the training dataset gets larger, which usually results in a long impractical training period. Moreover, the performance of triplet loss highly relies on the mining of hard triplets, which is also time consuming. Meanwhile, how to define "good" hard triplets is still an open problem. All factors above make triplet loss hard to train. In order to overcome these limitations of triplet loss, we will integrate it with center loss (see Sec. 3.2) and propose a novel TCL loss.

3.2. Review on center loss

Center loss [39] was proposed to compensate for softmax loss in face verification. It learns a center for the features of each class and meanwhile tries to pull the deep features of the same class close to the corresponding center as illustrated in Fig. 2 (b). Basically, center loss can be formulated as:

$$L_c = \frac{1}{2}\sum_{i=1}^{N} D\big(f(x^i), c_{y_i}\big) \tag{2}$$

where $c_{y_i} \in \mathbb{R}^d$ is the center of class y_i, with d denoting the dimension of features. Function $D(\cdot)$ stands for the squared Euclidean distance. During training, center loss encourages instances of the same classes to be closer to a learnable class center. However, since the parametric centers are updated at each iteration based on a mini-batch instead of the whole dataset, which is very unstable, it has to be under the joint supervision of softmax loss during training.

3.3. The proposed triplet-center loss

Motivation. Though the joint supervision of center loss and softmax loss aims at minimizing the intra-class variations and achieves very promising performances on face recognition, however, as illustrated in Fig. 2 (b), even though the intra-class variations are very small, the inter-class clusters are very likely overlapped. This is due to that it does not consider the separability of inter-class explicitly. While for triplet loss, it directly optimizes the network for the final task but subjects to the complexity of the construction of triplets. Motivated by the two representative losses, we bring up the triplet-center loss for the sake of learning more discriminative features efficiently.

Definitions. The goal of TCL is to leverage the advantages of triplet loss and center loss, *i.e.*, to efficiently minimize the intra-class distances of the deeply-learned features as well as maximize the inter-class distances of the deep features simultaneously. Let the given training dataset $\{(x^i, y^i)\}_{i=1}^N$ consists of N samples $x^i \in \mathcal{X}$ with the associated labels $y^i \in \{1, 2, ..., |\mathcal{Y}|\}$. And these samples are embedded into d-dimensional vectors with a neural network denoted by $f_\theta(\cdot)$. In TCL, we assume that the features of 3D shapes from the same class share one corresponding center. Thus we can obtain $\mathcal{C} = \{c_1, c_2, ..., c_{|Y|}\}$, where $c_y \in \mathbb{R}^d$ denotes the center vector for samples with label y, and $|\mathcal{Y}|$ is the number of centers. For simplicity, we adopt f_i to represent $f(x^i)$ in the following paper. Similar to center loss, we update the parametric centers at each iteration based on a mini-batch. Given a batch of training data with M samples, we define TCL as

$$L_{tc} = \sum_{i=1}^{M} \max\Big(D\big(f_i, c_{y^i}\big) + m - \min_{j \neq y^i} D\big(f_i, c_j\big), 0\Big) \tag{3}$$

where $D(\cdot)$ represents the squared Euclidean distance function denoted as:

$$D(f_i, c_{y^i}) = \frac{1}{2}||f_i - c_{y^i}||_2^2 \tag{4}$$

As illustrated in Fig. 2 (c), TCL is to push the distances between the samples and their corresponding center c_{y^i} closer than the distances between the samples and their nearest *negative center* (*i.e.*, centers of the other classes $\mathcal{C} \setminus \{c_{y^i}\}$) by a margin m.

To compute the back-propagation gradients of the input feature embeddings and the corresponding centers, we assume the following notations for demonstration: $\mathbb{1}[$*condition*$]$ is an indicator function which outputs 1 if the *condition* is satisfied and outputs 0 otherwise, $q_i = \operatorname{argmin}_{j \neq y^i} D\big(f_i, c_j\big)$ is an integer index which indicates the nearest negative center of i-th sample, and $\tilde{L}_i$ represents the triple-center loss of i-th sample as

$$\tilde{L}_i = \max\Big(D\big(f_i, c_{y^i}\big) + m - \min_{j \neq y^i} D\big(f_i, c_j\big), 0\Big) \tag{5}$$

Then, the derivatives of our TCL loss Eq. 3 with respect to the feature embedding of i-th sample $\frac{\partial L_{tc}}{\partial f_i}$ and j-th center $\frac{\partial L_{tc}}{\partial c_j}$ can be calculated as follows:

$$\begin{aligned}\frac{\partial L_{tc}}{\partial f_i} &= \Big(\frac{\partial D\big(f_i, c_{y^i}\big)}{\partial f_i} - \frac{\partial D\big(f_i, c_{q_i}\big)}{\partial f_i}\Big) \cdot \mathbb{1}[\tilde{L}_i > 0] \\ &= (c_{q_i} - c_{y^i}) \cdot \mathbb{1}[\tilde{L}_i > 0]\end{aligned} \tag{6}$$

$$\frac{\partial L_{tc}}{\partial c_j} = \frac{\sum_{i=1}^{M}(f_i - c_j) \cdot \mathbb{1}[\tilde{L}_i > 0] \cdot \mathbb{1}[y^i = j]}{1 + \sum_{i=1}^{M} \mathbb{1}[\tilde{L}_i > 0] \cdot \mathbb{1}[y^i = j]} - \frac{\sum_{i=1}^{M}(f_i - c_j) \cdot \mathbb{1}[\tilde{L}_i > 0] \cdot \mathbb{1}[q_i = j]}{1 + \sum_{i=1}^{M} \mathbb{1}[\tilde{L}_i > 0] \cdot \mathbb{1}[q_i = j]} \tag{7}$$

Joint supervision with softmax loss. Softmax loss focuses on mapping the samples to discrete labels, while TCL aims to apply metric learning to the learned embeddings directly. Unlike center loss, TCL can be used independently from softmax loss. However, these two losses can also be combined together to achieve more discriminative and robust embeddings according to our experiments in Sec. 4, which can be written as

$$L_{total} = \lambda L_{tc} + L_{softmax} \tag{8}$$

where λ is a hyper-parameter which controls the trade-off between the TCL and softmax loss. We attribute the benefit brought by softmax loss to the fact that the parametric centers of TCL are randomly initialized and updated based-on the mini-batches instead of the whole datasets which might be tricky, while softmax loss could serve as a good guider for seeking better class centers.

3.4. Discussion

Compared with triplet loss. Different from triplet loss, whose triplet is made up of triple samples (x_a^i, x_+^i, x_-^i), the triplet of the TCL is composed of the i-th sample x^i, its corresponding center c_{y^i} and its nearest *negative center*. For a training dataset with N samples, only N triplets will be formed for TCL while the number of triplets for triplet loss is $O(N^3)$ much more than TCL. Consequently, in comparison with triplet loss, TCL avoids the complexity of constructing triplets and the necessity for mining hard samples. We provide an empirical analysis on the two losses and visualize the learned embeddings using t-SNE in Sec. 4.1.

Compared with center loss. TCL can be taken as a variant of triplet loss, which could be exploited as supervision for training the neural networks independently of softmax loss, while center loss has to be combined with softmax to make the learning feasible otherwise the deeply learned features and centers will degrade to zeros according to [39]. Furthermore, TCL simultaneously maximizes the intra-class compactness and inter-class separability explicitly while center loss neglects the latter one which may lead to inter-class overlapping. Besides, center loss aims at reducing the absolute distances between the samples and their corresponding centers, while TCL penalizes the relative distances with a hinge-style loss which is more relaxed and easier to train. For empirical analysis of TCL loss and center loss, see Sec. 4.1 for details.

4. Experiments

In this section, we evaluate the performance of the proposed TCL on two kinds of 3D shape retrieval tasks: *generic 3D shape retrieval task* and *sketch-based 3D shape retrieval task*. The former is a within-domain retrieval task, where the given query and examples in the database are 3D models. While the latter is a cross-domain retrieval task, the given query are 2D sketches instead.

4.1. Generic 3D shape retrieval task

Datasets. We evaluate our TCL loss on two well-known 3D shape benchmarks: ModelNet40 [40] and ShapeNet Core 55 [27]. The ModelNet40 dataset contains 12,311 CAD models from 40 common categories. For this dataset, we follow previous works [33, 1] on the training and testing split settings, *i.e.*, randomly selecting 100 models per category, of which 80 models are used as training data and the rest for testing. The ShapeNet Core 55 dataset [27] is composed of 51,190 3D shapes in total from 55 categories, which are further divided into 204 sub-categories. Due to its diversity of categories and large variations within classes, ShapeNet55 dataset is quite challenging. The whole dataset is split into the training set, the validation set and the test set, which contains 35,765, 5,159, 10,266 models, respectively. Further, this dataset has two variants (ShapeNet 55 perturbed dataset and ShapeNet 55 normal dataset). For the normal dataset, shapes are aligned. For perturbed version of the dataset, each model is randomly rotated by an angle. We conduct experiments on the perturbed dataset which is more challenging. All retrieval performances are reported on the testing set.

Implementation Details. Experimental codes are implemented in PyTorch `http://pytorch.org/` and executed on a server with four Nvidia Titan-X GPUs, an Intel i7 CPU and 64GB RAM. We choose the Imagenet-pretrained VGG-A with batch normalization [32] [2] as our base network. VGG-A contains 8 convolution layers (*conv 1-8*) with kernels of size 3 by 3 and 3 fully-connected layers (*fc 9-11*). And the view-pooling layer is placed right after *conv 6*. The layers before and after the view-pooling layer are denoted as CNN1 and CNN2 respectively. We initialize the centers with a Gaussian distribution, and the mean and standard deviation is (0, 0.01) respectively. For optimization, we adopt the stochastic gradient descent algorithm with a mini-batch size of 16 for all our experiments. The learning rate for optimizing CNN1 and CNN2 is set to be 1e-4 and 1e-3 respectively. And for both CNN1 and CNN2, a weight decay of 1e-4 and a momentum of 0.9 are used. The learning rate

[2]Different from the original MVCNN, which adopts VGG-M as their base network, we adopt VGG-A with batch normalization to validate our proposed loss due to the lack of pre-trained VGG-M in Pytorch.

for centers is set to 0.1 for our experiments. We clip the gradient of the center by 0.01. During testing, We extract the output of *fc-10*, which is 4096-dimensional, as the features for all our retrieval tasks.

Parameter influence. As indicated by loss function in Eq. (8), the margin parameter m and λ may affect the final combinations of the losses. Specifically, λ in Eq. (8) controls the trade-off between softmax loss and TCL loss, while m controls the relative distance between the sample embeddings to its corresponding center and to its nearest negative center. To study the impact of the two hyper-parameters, we give an empirical analysis on ModelNet40 dataset.

The influence of hyper-parameter λ is presented in Fig. 3 (a). From the experimental results, we can see that TCL is very robust to this parameter. For a wide-ranged values from 0.01 to 10, the trained models consistently achieves very promising results on ModelNet40 dataset. We assume that this is because TCL and softmax loss are complementary losses. TCL focuses on the feature representation directly, while softmax loss focuses on how to separate or map the feature representation into a discrete label space. Once the model with softmax loss has converged, TCL can further enforce the feature embeddings into more compact clusters without sacrificing the classification performance too much, thus good retrieval performance can be gained, and vise versa. When λ is set to be 0, which means the model is trained using only softmax loss, the performance is worst, only achieving an mAP of nearly 80.0%. But with TCL, we can get an improvement of 7∼8% in terms of mAP. However, TCL loss is more sensitive to the margin m. To study the influence of m, we fixed λ to be 0.01, and then set m to be 0.5, 1, 5 and 10 respectively. For hyper-parameter m, too large or too small values both lead to inferior results. When it is too small, the strength of triplet center loss will be weakened, while too large value may cause over-fitting problem. The best results are achieved by setting m to be 5. This gives an mAP of 88.0% and an AUC of 89.0%. Thus, we set m and λ to be 5 and 0.01 in default respectively for the following experiments.

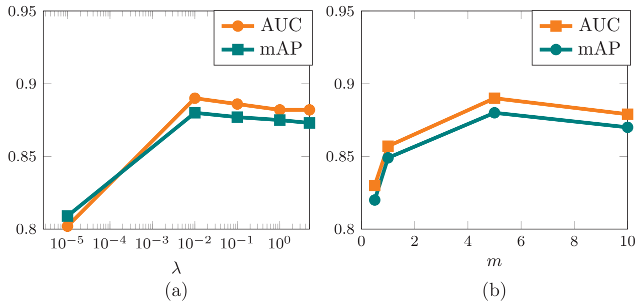

Figure 3. The retrieval performances achieved by (a) varying λ when m is fixed to 5 and (b) varying m when λ is fixed to 0.01.

Table 1. The performances (%) of different losses on ModelNet40.

Loss function	AUC	MAP
softmax loss	80.9	80.2
softmax+center loss	84.3	83.5
triplet loss	80.9	80.0
TCL	87.6	86.7
TCL+softmax loss	**89.0**	**88.0**

Comparison with other losses. To validate the proposed TCL loss, we conduct extensive experiments on various losses, including triplet loss, softmax loss, softmax loss with center loss, TCL and softmax loss with TCL on ModelNet40 dataset. As can be seen from Tab. 1, TCL and softmax loss with TCL perform best among these losses, obtaining an mAP of 86.7% and 88.0% respectively. In addition, softmax loss with center loss can increase the mAP by 3.4% over softmax loss, reaching 83.5%. The results of triplet loss are merely comparable to softmax loss.

Visualization of learned representations. We adopt t-SNE [20] to visualize the deeply-learned features of the samples from ModelNet40 dataset. As is shown in Fig. 4, some nice properties can be observed: (i) Compared with softmax loss, the learned embeddings of the same class are obviously getting closer to each other after introducing center loss, while triplet loss also results in slightly better embeddings since the sample embedding distances are considered directly. (ii) The proposed TCL performs better than center loss on achieving small intra-class variance and large inter-class variance. (iii) With the combination of TCL + softmax loss, learned embeddings make the most compact and separated clusters at the same time. This demonstrates the intuition that better underlying representations for retrieval task can be obtained with the proposed TCL loss.

Comparison with the state-of-the-arts. The performance comparisons with the state-of-the-art shape retrieval methods are presented in Tab. 2. Here we choose view-based retrieval methods including MVCNN [33], GIFT [1], DeepPano [31], and voxel-based retrieval methods including 3DShapeNet [22], DLAN [11] for comparison. As is shown, compared with MVCNN, which is firstly trained with softmax loss and then an off-line large-margin metric learning algorithm is applied, our method (TCL+softmax loss) has demonstrated its superior discriminative ability and improved mAP by nearly 7%, reaching 88.0%. Furthermore, we outperform GIFT by nearly 6% in terms of AUC and mAP respectively. And compared with DLAN, which is the current state-of-the art method on ModelNet40, an improvement of 3% in terms of mAP is obtained. Besides, our method also beats a recent re-ranking algorithm named Regularized Ensemble Diffusion (RED) [3]. Fig. 5 shows the representative retrieval results of our method on the ModelNet40 dataset.

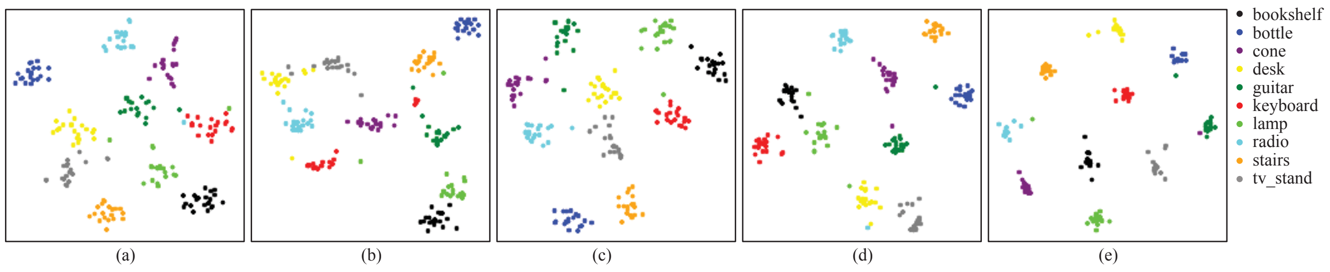

Figure 4. A visualization of deeply-learned features by (a) softmax loss, (b) triplet loss, (c) softmax loss + center loss, (d) triplet center loss, (e) softmax loss + triplet center loss. Here we randomly select 10 classes from the test set for visualization (Best viewed in color).

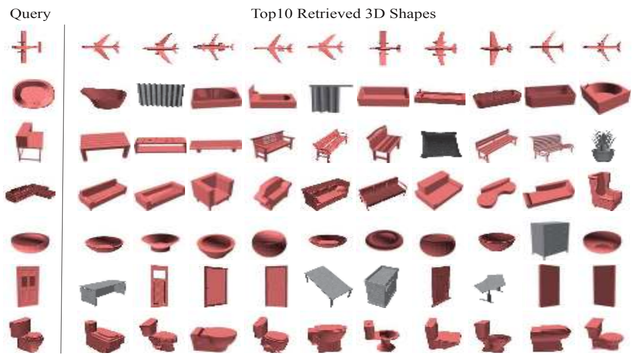

Figure 5. Retrieval examples on ModelNet40 dataset. Top matches are shown for each query with mistakes highlighted in gray.

Table 2. The performance (%) comparison with state-of-the-art methods on ModelNet40. '-' represents missing metric.

Methods	AUC	MAP
3DShapeNet [22]	49.9	49.2
DeepPano [31]	77.6	76.8
MVCNN [33]	-	80.2
GIFT [1]	83.1	81.9
DLAN [11]	-	85.0
RED [3]	87.0	86.3
Ours	**89.0**	**88.0**

Experiments on ShapeNet Core55. We evaluate our proposed TCL on the more challenging ShapeNet Core55 dataset. Since each shape in the dataset is attached with a category label and a subcategory label. To accurately evaluate the more fine-grained retrieval results, the organizer of SHREC'16 uses a metric named normalized discounted cumulative gain (NDCG). In addition, other traditional evaluation metrics like F-measure and mAP are also used.

The comprehensive comparisons with the state-of-the-art methods on the perturbed dataset are listed in Tab. 3, which is extremely challenging for the existing methods. We choose the following methods for comparisons, including GIFT [1], MVCNN [33], Wang [27], Li [27], Kd-network [15]. As is shown, our method (TCL+softmax loss) outperforms the strong baseline MVCNN by nearly 10% in terms of the micro-averaged mAP and 12% for the macro-averaged mAP. For other metrics, we also outperform by a large margin. Further, compared with GIFT, the former state-of-the-art in the contest, notable improvements are also achieved. These results demonstrate that the proposed loss is good at obtaining robust and discriminative representations for 3D shapes once again.

Table 3. The performance (%) comparison on ShapeNet55 test perturbed.

Methods	microALL			macroALL		
	F1	mAP	NDCG	F1	mAP	NDCG
MVCNN [33]	61.2	73.4	84.3	41.6	66.2	79.3
GIFT [1]	66.1	81.1	88.9	42.3	73.0	84.3
Wang [27]	24.6	60.0	77.6	16.3	47.8	69.5
Li [27]	53.4	74.9	86.5	18.2	57.9	76.7
Kd-network [15]	45.1	61.7	81.4	24.1	48.4	72.6
Ours	**67.9**	**84.0**	**89.5**	**43.9**	**78.3**	**86.9**

4.2. Sketch-based 3D shape retrieval task

Considering the huge visual gap between sketch and 3D shape, sketch-based shape retrieval is a quite challenging problem. Here we further demonstrate the potential advantages of TCL in such a cross-domain retrieval task.

Datasets. Two widely adopted sketch-based shape retrieval benchmarks have been adopted to evaluate our proposed method, they are SHREC'13 and SHREC'14 sketch track benchmark datasets respectively. SHREC'13 is composed of 7,200 sketches and 1,258 shapes, which are grouped into 90 classes. There are 80 sketches for each class, of which 50 sketches for training and 30 for testing. SHREC'14, which is the largest sketch-based shape retrieval dataset currently, contains 13,680 sketches and 8,987 shapes, divided into 171 classes. Due to its large variations within category and diversity of classes, SHREC'14 dataset is very challenging. For each class of sketches, 50 sketches are selected for training and 30 for testing.

Implementation Details. As for the experimental setup, we follow the same training-testing split as [17, 41]. Similar to [41], we render 12 views for each shape and employ

Table 4. The performance (%) comparison on SHREC13 dataset

Methods	NN	FT	ST	E	DCG	mAP
CDMR [10]	27.9	20.3	29.6	16.6	45.8	25.0
SBR-VC [17]	16.4	9.7	14.9	8.5	34.8	11.6
Siamese [35]	40.5	40.3	54.8	28.7	60.7	46.9
DCML [9]	65.0	63.4	71.9	34.8	76.6	67.4
LWBR [41]	71.2	72.5	78.5	36.9	81.4	75.2
Ours	**76.3**	**78.7**	**84.9**	**39.2**	**85.4**	**80.7**

two AlexNets [16] to extract features of 2D projections and sketches, respectively. MVCNN framework is adopted to obtain the representations of 3D shapes. In addition, we assume that the features from these two different domains share the same centers and classifiers. Thus we could train the two AlexNets pre-trained on ImageNet dataset jointly via back propagation with TCL and softmax loss. The parameters are given as λ = 0.01, m = 5.0. The other settings are the same as Sec. 4.1. To measure the retrieval performances, we use the following metrics including Nearest Neighbor (NN), First Tier (FT), Second Tier (ST), E-measure (E), Discounted Cumulated Gain (DCG) and mean Average Precision (mAP).

Experiments on SHREC'13 and SHREC'14. In Tab. 4 and Tab. 5, a comprehensive comparison with various state-the-of-arts is presented, including Sketch-Based Retrieval method with View Clustering (SBR-VC) [17], Cross Domain Manifold Ranking method (CDMR) [10], Siamese network (Siamese) [35], Deep Correlated Metric Learning (DCML) [9] and Learned Wasserstein Barycentric Representations (LWBR) [41]. Among them, DCML and LWBR perform best. DCML [9] proposes discriminative loss and correlation loss, aiming to increase the discrimination of features within each domain as well as the correlation between different domains. While LWBR [41] mainly focuses on improving 3D shape representations which proposes to adopt wasserstein barycenters of features of multiple views as 3D shape representations instead of max-pooling in MVCNN [33]. These methods have advanced other methods a lot thanks to the powerful of deep learning. Especially LWBR gets 75.2% mAP on SHREC'13, nearly 8% higher than DCML. However, benefit from TCL, more discriminative features are obtained. Even without sophisticated computation of barycentric representations, our method (TCL+softmax loss) achieves 79.8% mAP on SHREC'13 dataset, 46.2% mAP on SHREC'14 dataset, which outperforms LWBR with at least 5% on both datasets.

4.3. View-based v.s. Model-based

So far we have demonstrated the superiority of the proposed TCL for view-based and sketch-based shape retrieval tasks. Actually, a sketch can be considered as a "special" view of the object, which is why we train them jointly with the projected views and got notable rewards in the retrieval. Whereas it has been proved that deeply-learned features from model-based approaches are complementary with those of view-based methods, we still conduct a comparison by further combining TCL into two representative model-based approaches: PointNet [24] and VoxNet [21]. Experiments are performed on ModelNet40 dataset. As we can see in Tab. 6, though less impressive compared to view-based mechanisms, significant improvements of 3%$\sim$4% are gained compared with baseline methods. Nevertheless, we believe that better performance can be achieved once the baseline of model-based deep learning approaches is improved. We leave it as a future work.

Table 5. The performance (%) comparison on SHREC14 dataset

Methods	NN	FT	ST	E	DCG	mAP
CDMR [10]	10.9	5.7	8.9	4.1	32.8	5.4
SBR-VC [17]	9.5	5.0	8.1	3.7	31.9	5.0
Siamese [35]	23.9	21.2	31.6	14.0	49.6	22.8
DCML [9]	27.2	27.5	34.5	17.1	49.8	28.6
LWBR [41]	40.3	37.8	45.5	23.6	58.1	40.1
Ours	**58.5**	**45.5**	**53.9**	**27.5**	**66.6**	**47.7**

Table 6. The performance (%) comparison on ModelNet40 using VoxNet and PointNet.

Methods	VoxNet		PointNet	
	AUC	MAP	AUC	MAP
softmax loss	70.9	70.0	71.4	70.5
softmax loss + center loss	73.1	72.0	74.3	73.1
TCL loss	73.4	72.4	72.3	71.3
TCL + softmax loss	**74.3**	**73.2**	**75.6**	**74.5**

5. Conclusion

In this paper, we focus on the view-based 3D shape retrieval task and propose a novel loss function named triplet center loss. This loss function combines the advantages of triplet loss and center loss, and has demonstrated its effectiveness on both 3D shape retrieval task and sketch based shape retrieval task. The triplet center loss can optimize the features directly by minimizing the intra-class variance while also maximizing the inter-class variance at the same time. As a result, the learned embeddings are more robust and discriminative, thus more appropriate for retrieval task. In the future, we would like to explore our proposed loss to other retrieval tasks such as person re-identification, face recognition and content-based image retrieval.

Acknowledgements This work was supported by NSFC 61573160 and 61602461, to Dr. Xiang Bai by the National Program for Support of Top-notch Young Professionals and the Program for HUST Academic Frontier Youth Team.

References

[1] S. Bai, X. Bai, Z. Zhou, Z. Zhang, and L. Jan Latecki. Gift: A real-time and scalable 3d shape search engine. In *CVPR*, pages 5023–5032, 2016.

[2] S. Bai, X. Bai, Z. Zhou, Z. Zhang, Q. Tian, and L. J. Latecki. Gift: Towards scalable 3d shape retrieval. *TMM*, 19(6):1257–1271, 2017.

[3] S. Bai, Z. Zhou, J. Wang, X. Bai, L. Jan Latecki, and Q. Tian. Ensemble diffusion for retrieval. In *ICCV*, 2017.

[4] S. Biasotti, A. Cerri, M. Abdelrahman, M. Aono, A. B. Hamza, M. El-Melegy, A. Farag, V. Garro, A. Giachetti, D. Giorgi, et al. Shrec14 track: Retrieval and classification on textured 3d models. In *3DOR*, pages 111–120, 2014.

[5] D. Boscaini, J. Masci, E. Rodolà, and M. Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *NIPS*, pages 3189–3197, 2016.

[6] D. Boscaini, J. Masci, E. Rodolà, M. M. Bronstein, and D. Cremers. Anisotropic diffusion descriptors. *Computer Graphics Forum*, 35(2):431–441, 2016.

[7] A. X. Chang, T. A. Funkhouser, L. J. Guibas, P. Hanrahan, Q.-X. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. *arXiv:1512.03012*, 2015.

[8] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, volume 1, pages 539–546, 2005.

[9] G. Dai, J. Xie, F. Zhu, and Y. Fang. Deep correlated metric learning for sketch-based 3d shape retrieval. In *AAAI*, pages 4002–4008, 2017.

[10] T. Furuya and R. Ohbuchi. Ranking on cross-domain manifold for sketch-based 3d model retrieval. In *Cyberworlds (CW)*, pages 274–281. IEEE, 2013.

[11] T. Furuya and R. Ohbuchi. Deep aggregation of local 3d geometric features for 3d model retrieval. In *BMVC*, 2016.

[12] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *arXiv:1703.07737*, 2017.

[13] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris. Deep learning advances in computer vision with 3d data: A survey. *ACM Computing Surveys (CSUR)*, 50(2):20, 2017.

[14] E. Johns, S. Leutenegger, and A. J. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *CVPR*, pages 3813–3822, 2016.

[15] R. Klokov and V. Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. *arXiv:1704.01222*, 2017.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012.

[17] B. Li, Y. Lu, A. Godil, T. Schreck, M. Aono, H. Johan, J. M. Saavedra, and S. Tashiro. Shrec'13 track: Large scale sketch-based 3d shape retrieval. In *Eurographics Workshop*, pages 89–96, 2013.

[18] Y. Li, S. Pirk, H. Su, C. R. Qi, and L. J. Guibas. Fpnn: Field probing neural networks for 3d data. In *NIPS*, pages 307–315, 2016.

[19] H. Liu, Y. Tian, Y. Yang, L. Pang, and T. Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *CVPR*, pages 2167–2175, 2016.

[20] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *JMLR*, 9(Nov):2579–2605, 2008.

[21] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015.

[22] A. Notchenko, E. Kapushev, and E. Burnaev. Sparse 3d convolutional neural networks for large-scale shape retrieval. *arXiv:1611.09159*, 2016.

[23] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, pages 4004–4012, 2016.

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[25] 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 *CVPR*, pages 5648–5656, 2016.

[26] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, 2017.

[27] M. Savva, F. Yu, H. Su, M. Aono, B. Chen, D. Cohen-Or, W. Deng, H. Su, S. Bai, X. Bai, et al. Shrec16 track: large-scale 3d shape retrieval from shapenet core55. In *3DOR*, 2016.

[28] M. Savva, F. Yu, H. Su, A. Kanezaki, T. Furuya, R. Ohbuchi, Z. Zhou, R. Yu, S. Bai, X. Bai, et al. Shrec17 track large-scale 3d shape retrieval from shapenet core55. In *3DOR*, 2017.

[29] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015.

[30] N. Sedaghat, M. Zolfaghari, and T. Brox. Orientation-boosted voxel nets for 3d object recognition. In *BMVC*, 2017.

[31] B. Shi, S. Bai, Z. Zhou, and X. Bai. Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343, 2015.

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014.

[33] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, 2015.

[34] J. W. Tangelder and R. C. Veltkamp. A survey of content based 3d shape retrieval methods. *Multimedia tools and applications*, 39(3):441, 2008.

[35] F. Wang, L. Kang, and Y. Li. Sketch-based 3d shape retrieval using convolutional neural networks. In *CVPR*, pages 1875–1883, 2015.

[36] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille. Normface: L 2 hypersphere embedding for face verification. In *ACM on Multimedia Conference*, pages 1041–1049. ACM, 2017.

[37] J. Wang, F. Zhou, S. Wen, X. Liu, and Y. Lin. Deep metric learning with angular loss. In *ICCV*, pages 4321–4329, 2017.

[38] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-cnn: octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017.

[39] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, pages 499–515, 2016.

[40] 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*, pages 1912–1920, 2015.

[41] J. Xie, G. Dai, F. Zhu, and Y. Fang. Learning barycentric representations of 3d shapes for sketch-based 3d shape retrieval. In *CVPR*, pages 5068–5076, 2017.

[42] J. Xie, G. Dai, F. Zhu, E. K. Wong, and Y. Fang. Deepshape: Deep-learned shape descriptor for 3d shape retrieval. *TPAMI*, 39(7):1335–1345, 2017.

[43] F. Zhu, J. Xie, and Y. Fang. Learning cross-domain neural networks for sketch-based 3d shape retrieval. In *AAAI*, pages 3683–3689, 2016.