DeepCCFV: Camera Constraint-Free Multi-View Convolutional Neural Network for 3D Object Retrieval

Zhengyue Huang,[1] Zhehui Zhao,[1] Hengguang Zhou,[2] Xibin Zhao,[1*] Yue Gao[1*]
[1]BNRist, KLISS, School of Software, Tsinghua University, China.
[2]Department of Computer Science, University of Toronto.
{hzy18, zhao-zh16}@mails.tsinghua.edu.cn, dsvsddr@gmail.com, {zxb, gaoyue}@tsinghua.edu.cn

Abstract

3D object retrieval has a compelling demand in the field of computer vision with the rapid development of 3D vision technology and increasing applications of 3D objects. 3D objects can be described in different ways such as voxel, point cloud, and multi-view. Among them, multi-view based approaches proposed in recent years show promising results. Most of them require a fixed predefined camera position setting which provides a complete and uniform sampling of views for objects in the training stage. However, this causes heavy over-fitting problems which make the models failed to generalize well in free camera setting applications, particularly when insufficient views are provided. Experiments show the performance drastically drops when the number of views reduces, hindering these methods from practical applications. In this paper, we investigate the over-fitting issue and remove the constraint of the camera setting. First, two basic feature augmentation strategies *Dropout* and *Dropview* are introduced to solve the over-fitting issue, and a more precise and more efficient method named *DropMax* is proposed after analyzing the drawback of the basic ones. Then, by reducing the over-fitting issue, a camera constraint-free multi-view convolutional neural network named *DeepCCFV* is constructed. Extensive experiments on both single-modal and cross-modal cases demonstrate the effectiveness of the proposed method in free camera settings comparing with existing state-of-the-art 3D object retrieval methods.

Introduction

3D object retrieval received growing attention in the field of computer vision since the rapid development of 3D applications in recent years. Various methods based on different 3D object representations, such as voxel, point cloud, and multi-view, have been investigated (Maturana and Scherer 2015; Qi et al. 2017a; Su et al. 2015). In multi-view based approaches, views can be easily captured by cameras or rendered from 3D object models, and these approaches showed promising results. We noticed that most of the approaches require a fixed predefined camera position setting which provides a full and uniform sampling of views for objects in the training stage. However, in practical applications, camera positions are randomly selected, the information patterns are different in the training stage and real applications, and it brings the pattern gap as illustrated in Fig.1. This might cause heavy over-fitting problems.

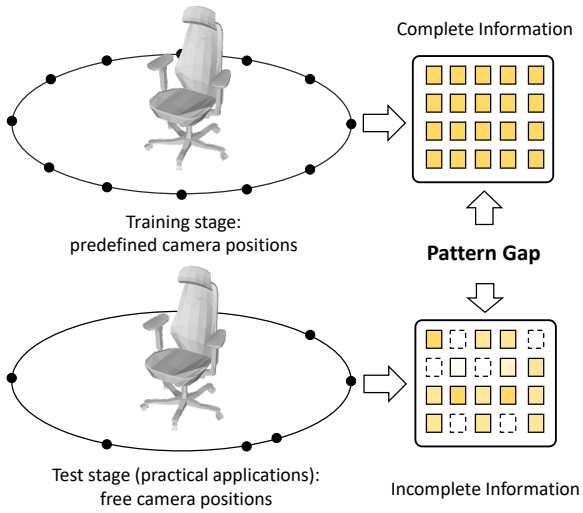


Figure 1: The pattern gap between training and test stage.

One typical multi-view based method is Multi-view Convolutional Neural Networks (MVCNN) (Su et al. 2015), which fused the 2D CNN features extracted from rendered views captured in a predefined camera setting in both training and test stage. Since then, more multi-view based neural network methods (Feng et al. 2018; Kanezaki, Matsushita, and Nishida 2018) have been introduced for 3D shape recognition and retrieval, which assumed a predefined camera setting in the test stage. However, the requirement for fixed-camera setting in test stage is impractical for real recognition tasks, in which the target 3D object may contain any number of views captured in any position, and it could be impossible to obtain the views from predefined directions. Providing insufficient views to MVCNN-like networks at test stage leads to huge performance drop as the number of views reduces, which was demonstrated in the experi-

*Corresponding authors

ments of GVCNN (Feng et al. 2018). It is reasonable that the performance drops since the input information is partially missed. However, an experiment in MVCNN revealed that training plain 2D image CNN for all captured views without features fusion achieves much better performance comparing to the original MVCNN when testing with single input view. This suggests that deep learning method itself have the potential to handle the single-view case, and this motivates us to make further studies on this issue.

We noticed these works following MVCNN require a predefined camera setting, which is convenient at the training stage to gather a full and uniform sampling from views of objects. However, this may cause heavy over-fitting problems since (1) Views provided in real applications may be different on camera position in training data (2) Number of views provided in real applications may be insufficient. Since the predefined camera positions usually sample densely enough to cover all positions, we focus on the second over-fitting problem in this paper. In (Gao et al. 2012), a camera constraint-free view-based 3D object retrieval method (CCFV) has been introduced to handle this situation using a probabilistic approach. With the development of deep learning methods in the field of 3D object retrieval and recognition, existing methods meet new challenges.

As discussed before, the number of views provided in actual applications may be insufficient which leads to the pattern gap between test stage and training stage. Therefore training on fixed camera setting data may lead to over-fitting problems and fail to adapt to the gap. In the training stage, the complete set of information results in the domination of some strong features in the network. To tackle this problem, we first introduce *Dropout*, a widely used feature augmentation method to reduce the domination of some strong features. On the other hand, *Dropview* (Nitish et al. 2014) is introduced which randomly drops the input views and tries to minimize the pattern gap between training and test stage. Both of the methods are proved useful by experiments with decent results. Moreover, We further propose *DeepCCFV*, which is a camera constraint-free multi-view convolutional neural network assembled with *DropMax* block which drops the highest stimulates of a network's layer output and forces the network to learn the relatively *weak* features. Thus the problem of domination of the strong features is more precisely located than *Dropout* and *Dropview*. Extensive experiments with different CNN backbones i.e. VGG11 (Simonyan and Zisserman 2014) with batch normalization and ResNet50 (He et al. 2016) showed that *DeepCCFV* has stable and superior performance than other methods in 3D object retrieval tasks. Finally, a trial on cross-modal retrieval is conducted using a common cross-modal pipeline to prove further that *DeepCCFV* can generate robust feature descriptors for 3D objects in multi-view representation. More details are provided in Section Experiments.

In summary, the main contributions of this paper are listed as follows:

- We propose a **Deep c**amera **c**onstraint-**f**ree multi-**v**iew neural network (DeepCCFV) for 3D object retrieval with a feature augmentation method called *DropMax*, dropping high stimulates of the output to force the model to pay more attention on less stimulative features, which tackles the over-fitting issue and brings notable improvement of retrieval performance.
- We conduct cross-modal 3D object retrieval trial between point cloud and multi-view modalities using the proposed *DeepCCFV* model.

The rest of this paper is organized as follows. Firstly, we introduce the related work on 3D object recognition, and then we provide a detailed explanation of our proposed method. After the explanation, experiments are presented.

Related Work

3D object retrieval and recognition tasks based on different data representations, such as voxel, point cloud, and multi-view, have been investigated in recent years. In this work, the proposed *DeepCCFV* is based on multi-view. Also, in the cross-modal retrieval trail, we adopt the point cloud representation as another modality input. Hence, we will briefly review the methods based on multi-view and point cloud in this section.

Multi-view Based Methods

Previous multi-view based methods studied and described 3D objects with a group of projected views from different angles of the object. One of the pioneer view-based methods is (Chen et al. 2003), which composed a group of ten views captured from the vertices of a dodecahedron over a hemisphere. In (Papadakis et al. 2009), a set of panoramic views are generated to represent the 3D objects. (Gao et al. 2012) proposed a general framework based on Gaussian models, which enabled eliminating fixed predefined camera setting using a probabilistic approach.

With the recent proliferation of deep learning, view-based models came into the spotlight. An instance used deep learning method, MVCNN (Su et al. 2015), extracted features with 2D CNN and then aggregated them through a pooling structure. Furthermore, Mahalanobis metric was employed to improve its performance on retrieval task. Input shapes are assumed to be upright oriented along a consistent axis, 12 rendered views are uniformly sampled around the object every 30 degrees. In (Xie et al. 2015), a deep auto-encoder structure was implemented to generate features of 3D objects. Based on MVCNN, GVCNN (Feng et al. 2018) divided extracted features into groups with different weights, followed by a weighted-average pooling. A deep embedding network was proposed to solve the complex intra-class and inter-class variation in (Guo et al. 2016), in which the deep neural network was jointly supervised by classification loss and triplet loss. RotationNet (Kanezaki, Matsushita, and Nishida 2018) treated viewpoints as latent variables; basis axes was determined with unsupervised learning on viewpoints estimation. Instead of preprocessing of normalizing object dataset, RotationNet learned pose alignment.

Following MVCNN, most of these works adopted a predefined camera setting for training convenience. However, this may cause a heavy over-fitting problem in the test stage in free camera settings. As experiments conducted in

GVCNN (Feng et al. 2018), the performances of MVCNN-like methods significantly decrease when the number of views drops. Due to the complex condition in a real-world environment, missing of views is common in applications, which limits the applications of view-based models in the real world. In this paper, we will propose methods to reduce the performance drop with insufficient views and accomplish camera-constraint-free in the next section.

Point Cloud Based Methods

Point cloud is regarded as an unordered set of 3-dimension vectors which denote the spatial positions of the captured object points. Recently, interest in point cloud gradually increases since the rapid development of sensors like LiDAR, which was widely adopted in auto-driving cars. While 3D recognition tasks based on point cloud is getting more and more attention, the cross-modal retrieval task between point cloud and multi-view is more or less neglected by the community, we made an initial trial on this topic using the proposed DeepCCFV as the feature extractor of multi-view modality, along with DGCNN for the point cloud modality.

For point cloud based methods, PointNet(Qi et al. 2017a) is one of the pioneer deep learning work which employed a spatial transform network to maintain the invariance of learned representation under transformation. In PointNet, a structure that aggregates local and global features was used to describe point-to-point correlation. The extracted features are processed through symmetric pooling layer to address the unorderedness of input. PointNet has achieved outstanding performance in both classification and segmentation tasks, yet the model learned local geometric features poorly. Hence, PointNet++ (Qi et al. 2017b) was introduced to address this flaw. PointNet++ built a hierarchical network structure with PointNet network as its basic building block. In (Klokov and Lempitsky 2017), the point cloud data was subdivided by Kd-tree, then features were extracted and aggregated by Kd-Network. By aggregating point and its neighbor features and perform edge convolution alternatively on extracted edge feature in the network, DGCNN(Wang et al. 2018) utilized both local features and global features. At the same time, the graph on each layer was dynamic which enabled observation over the semantically similar structure in the deeper layer of feature space. Among these methods, DGCNN was reported to reach the state-of-the-art performance in 3D object recognition. With DGCNN as the basic structure, PVNet (You et al. 2018) proposed an embedding attention fusion scheme, which integrated and utilized the high-level features of point cloud data and multi-view data to augment each other in 3D shape recognition. In the cross-modal retrieval trial, we adopted DGCNN as the point cloud feature extractor.

Proposed Method

3D object retrieval aroused a great deal of interest in the last few years. However, most of the previous view-based methods require a fixed setting of camera positions, which limits the application of these methods. Therefore, we propose *DeepCCFV* to address this issue. Extensive experiments have been conducted to prove its superior and stable performance in both intra-modal and cross-modal cases. In the cross-modal retrieval experiment, the widely used point cloud is adopted as the supplemental modality.

The Problem of Predefined Camera Setting

Most of the recent works based on multi-view representation of 3D objects are trained with rendered views captured in a predefined camera setting, which is convenient in the training stage to gather a full and uniform sampling of views for objects. However, in such cases the embedding space of the retrieval network is optimized by a complete set of features of 3D objects, some *strong* features will dominate the training process and *weak* features will not be learned. This will cause the over-fitting problems since the presentation of *strong* features in test stage is not guaranteed with free camera settings.

Tackle the Over-fitting Issue

Basic Feature Augmentation Methods We first introduce two basic feature augmentation methods *Dropout*(Nitish et al. 2014) and *Dropview* to overcome the over-fitting issue. *Dropout* is a widely used feature augmentation method which can reduce the domination of some strong features, but it drops *weak* features equally. *Dropview* is a simple data augmentation strategy introduced in this paper which can also reduce the domination of some strong features through randomly dropping the input views, but weaker features in dropped view are lost and cannot be further enhanced in the training process.

DropMax Considered the drawback of these methods, we propose *DropMax* which drops the $topk$ highest stimulates of the output layer of the network, and it forces the network to learn relatively *weak* features. For the MVCNN-like architecture, we add a *DropMax* block before and after the aggregation operation as illustrated in Fig.2. In our perspective, high stimulates correspond to *strong* features processes in a network since *strong* features contribute the most in both forward and backward processes which may dominate the training process. Thus dropping *strong* features instead of dropping randomly locates the issue of *strong* features domination more precisely.

Formally, *DropMax*, which drops the top-k highest stimulates at the probability of p, is added after a fully-connected layer of deep neural network after activation function. In the forward process, the dropped stimulates are set to zero and therefore zero gradient is passed through these dropped neurons in the backward process. This formula is illustrated as follows:

$$m^{(l)} = TopkMask(k, p) \tag{1}$$

$$\tilde{y}^{(l)} = (1 - m^{(l)}) * y^{(l)} \tag{2}$$

where $TopkMask$ represents a function to generate the $topk$ mask at probability p, l denotes the corresponding layer, $y^{(l)}$ is the original output of layer l, $\tilde{y}^{(l)}$ represents the masked output after DropMax. In a large range, the selection of hyper parameters k and p leads to stable performance, further discussion is shown in the Experiment Section.

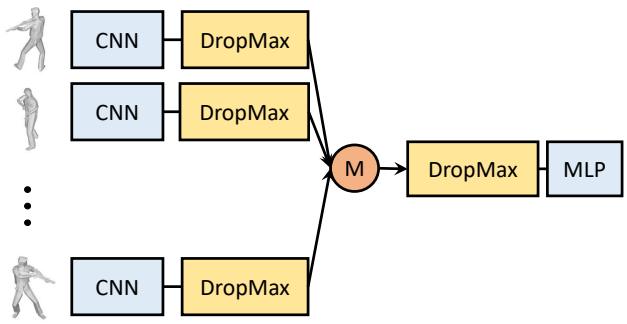


Figure 2: *DeepCCFV* architecture, DropMax block is added before and after the aggregation operation (denoted as M is this figure), which is the maxpooling layer in MVCNN.

DeepCCFV By reducing the over-fitting issue, which is the main problem that limits the generalization of current methods to free camera setting, a camera constraint-free multi-view convolutional neural network named *DeepCCFV* assembled with *DropMax* block is constructed.

DeepCCFV for Single-modal Retrieval

In single-modal retrieval task, we calculate the cross-entropy loss on the basic *DeepCCFV* network to obtain discriminative embedding features. To reach greater robustness in retrieval, we adopt a common l_2-normalization strategy applied on the raw features generated by the network (Wang et al. 2017; Song et al. 2016; Parkhi, Vedaldi, and Zisserman 2015; Schroff, Kalenichenko, and Philbin 2015). We conduct extensive comparison experiments to prove the effectiveness of *DeepCCFV* with other augmentation methods. Details are shown in the Experiment Section.

DeepCCFV for Cross-modal Retrieval

To further test the robustness of the proposed *DeepCCFV*, we provide an initial trial of cross-modal experiment between multi-view and point cloud modalities of 3D objects. For simplicity, we adopt a common pipeline for cross-modal retrieval tasks which is illustrated in Fig.3. The inputs of the retrieval network are the data of two modalities, point cloud, and multi-view. The point cloud is a set of unordered points densely sampled from the 3D objects' faces. The data of multi-view are rendered images of 3D shapes captured in predefined camera positions following the (Su et al. 2015) while the number of views is variable in this experiment.

Taking two modalities' input at the same time, the feature extractor networks first extract features of two models simultaneously, then pass the output features to the following embedding network that aims to learn a deep metric transformation for mapping the features of different modalities to the same embedding space. Finally, Triplet-Center-Loss (He et al. 2018) is adopted as the retrieval loss calculation on the embedded features in each mini-batch. Besides, for better performance, we consider different places to calculate the cross-entropy loss for each modality; placing it after mlp_1 layer in our embedding network leads to the most excellent performance. The whole pipeline is illustrated in Fig.3.

Experiments

In this section, we provide experiments on 3D object retrieval in both single-modal and cross-modal cases which demonstrated the better and more stable performance comparing with other methods particularly when the number of views is limited. At last, an additional experiment on the classification task is conducted which also showed the superior performance of the proposed method *DeepCCFV*.

Dataset

To validate the performance of the proposed *DeepCCFV*, we conduct experiments on Princeton ModelNet dataset (Chang et al. 2015), which contains 127,915 3D CAD models from 662 categories in total. We applied ModelNet40, a commonly used subset of ModelNet that contains 12,311 shapes from 40 common categories in our experiments. We follow the same training/test split of ModelNet40 as in MVCNN (Su et al. 2015).

Single-modal Retrieval

As discussed before, the over-fitting problem comes from two aspects: (1) camera positions of the query views are different from training data (2) number of query views provided is insufficient in the test stage. To validate if the proposed method helps the second dilemma and diminish the over-fitting problem we took the position of the cameras as the control variable and the quantity and selection of views as independent variables. Furthermore, to evaluate the performance of our proposed *DeepCCFV* network in the real-world situation, in which given queries are randomly captured in unfixed camera positions, we conduct the same experiment with views rendered in random angle around the 3D object. To give a fair comparison with the original MVCNN, the first camera setting in MVCNN, i.e., 12 predefined camera positions are adopted in the training stage.

To evaluate *DropMax*'s effectiveness, we compare the proposed *DeepCCFV* assembled with *DropMax* block with (1) Original MVCNN (2) MVCNN with *Dropout* (3) MVCNN with *DropView*. For details, *Dropout* is added at the same place as *Dropview* with the same drop strength. *DropView* is implemented by randomly dropping the input views and it is insensitive to the drop strength so we set it to 0.5 in the experiments.

Implementation In the predefined camera position setting, the views are rendered around the 3D models in every 30 degrees, 12 views for each object. While in the random camera position setting, the views are rendered around the 3D models in random angles, and the total number of views for each object is also 12.

In our experiments, two commonly-used CNN backbones, VGG11 (Simonyan and Zisserman 2014) with batch normalization(Ioffe and Szegedy 2015) and ResNet50 (He et al. 2016), are implemented as the image feature extractor in our experiments, which is slightly different from the original MVCNN (Su et al. 2015) that uses VGG-M. For a fair comparison, we re-implement MVCNN with these backbones. Both of them are pre-trained on ImageNet(Deng et al. 2009).

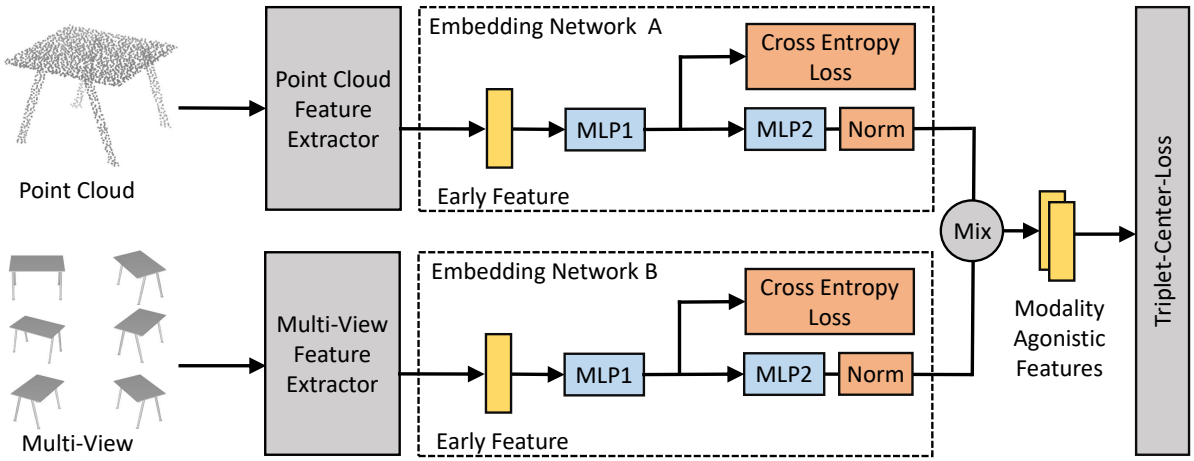


Figure 3: The architecture of our cross-modal retrieval network. The input multi-view data and point cloud data is first passed through the feature extractors. Then the extracted features are passed to the embedding networks to generate two features for each modality. These features are employed in both single-modal and cross-modal retrieval.

The result showed that using a stronger CNN backbone in MVCNN leads to better performance.

For the following experiments, the retrieval database contains the features of 3D objects in the test set split of ModelNet40 which are generated by the network with all 12 views from predefined positions provided. As for the queries, we use the number of randomly selected views of 3D objects ranges from 1 to 12. All these features are extracted from the output of fc_7 layer which is defined in MVCNN, right before the classifier.

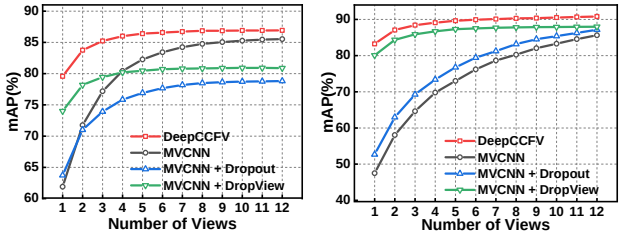
We rank the similarity of the features of queries and the features in the database by measuring the Euclidean distance between them, and the top 1000 results with the closest distance are selected as the retrieval output. Mean average precision (mAP) is then calculated on this output. For a fair comparison, mAP is calculated following the previous work (Chang et al. 2015; Su et al. 2015). Furthermore, since our query views are randomly selected, we carry out each experiment for ten times and calculate the average value of them.

Parameter Influences There are two hyperparameters in the *Dropmax* block, i.e., p and k. We evaluate the influences of parameters and select the parameters based on the experimental analysis.

In our evaluation, experiments demonstrated that the performance of our model is insensitive with regards to the value of p. It has stable performance when p ranges from 0.1 to 0.8. Hence, we empirically set p to 0.4 in both of the *DropMax* blocks.

To evaluate the parameter k, we denote k_1 for the first *DropMax* block and k_2 for the second. First, we fix k_2 to 50 and vary k_1 from 100 to 2000. We observed that our method could achieve stable results when k_1 varies in a large range from 300 to 600. Thus, we set k_1 to 500 in all experiments. To evaluate the parameter k_2 for the second *DropMax*, we fix k_1 to 500 and vary k_2 from 0 to 400. We observed that our method has stable performance when k_2 varies in a large range from 0 to 400, especially when k_2 varies from 0 to 100. We then set k_2 as 50 in all experiments. All these experiments demonstrated that the proposed method is insensitive to the value of the parameters k and p. Moreover, should be noticed that we only select the stable parameters in these experiments, the optimal parameters setting can be further studied in the future work.

Over-fitting Problem Diminution In this experiment, the query views are captured in predefined camera positions. This experiment evaluates the ability of *DeepCCFV* to diminish the over-fitting problem and improve retrieval performance in quantity-limited views situation. The experiment results and comparison among different methods are demonstrated in Fig.4(a) and Fig.4(b).



(a) VGG11-BN (b) ResNet50

Figure 4: The retrieval mAP of comparing approaches using VGG11-BN and ResNet50 as the CNN backbone in predefined camera positions. The proposed *DeepCCFV* had the best performance among the methods listed in the figure. The retrieval mAP of *DeepCCFV* using single view achieved 79.57% and 83.25%, which were 17.66% and 35.75% higher than basic MVCNN. The retrieval mAP of our proposed *DeepCCFV* model using all 12 views achieved 86.92% and 90.82% ,which were 1.40% and 5.18% higher than basic MVCNN.

In the experiment using VGG11-BN as the backbone as illustrated in Fig.4(a), the result of MVCNN demonstrated relatively decent result with 1.4% lower than *DeepCCFV* in the full-view setting. However, the difference between the result of MVCNN in a single-view setting and a full-view setting reached 23.61%, which is extreme. Similar to MVCNN, MVCNN with *Dropout* was also sensitive to the number of views, with 15.07% difference between the worst and the best cases. In contrast, MVCNN with *Dropview* and *DeepCCFV* were comparatively stable, with 6.89% and 7.35% difference between the worst and the best cases.

A similar pattern can be observed when using Resnet50 as the backbone illustrated in Fig.4(b). Curves of MVCNN and MVCNN with *Dropout* were lower than other methods, and their difference between the highest and lowest points were 38.14% and 34.47%, while curves of *DeepCCFV* and MVCNN with *Dropview* had the difference of 7.57% and 7.92%. This result implies that MVCNN with *Dropview* and *DeepCCFV* with *DropMax* as its core is more powerful in overcoming the over-fitting issue.

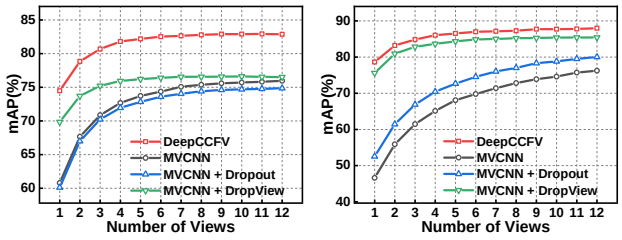
In both of the figures, we can see that the performance curve of MVCNN decreased sharply by dropping the number of views, while the performance curve of *DeepCCFV* is not entirely affected; the fewer views given in queries, the more significant difference is observed. Plain MVCNN suffered more from over-fitting in full-view training, resulting in a greater reduction in mAP because of incomplete feature sets under single-view setting. On the other hand, our purposed *DeepCCFV* outperformed other methods in every setting. All these experiments demonstrated the proposed method *DeepCCFV* could reduce the over-fitting problem effectively and generate more robust and more comprehensive features of multi-view representations of 3D objects.

Besides, we noticed an interesting phenomenon that plain MVCNN performs better than MVCNN with *Dropout* and MVCNN with *Dropview* when using VGG11-BN as the backbone, while MVCNN with *Dropout* and MVCNN with *Dropview* performs better when using ResNet50 as the backbone. As far as we concerned, this is because VGG11-BN model already employs *Dropout* layers, when extra *Dropout* layers are added or *Dropview* is added to drop the input of views, too much information is dropped, and it becomes a hinder for the model to learn discriminative features for 3D objects.

Free Camera Setting In this experiment, the query views are captured in random camera positions around the object. This experiment evaluates the performance of *DeepCCFV* in a real-world situation. The experiment results and comparison among different methods are demonstrated in Fig.5(a) and Fig.5(b).

In Fig.5(a), MVCNN, MVCNN with *Dropout* and MVCNN with *Dropview* achieved closed highest result of 75.96%, 74.86%, and 76.52%, while the proposed *DeepCCFV* reached outstanding highest result of 82.87%.

Just like results in Fig.4(b), in Fig.5(b), MVCNN and MVCNN with *Dropout* had similar results of 76.23% and 80.03% under full-view setting and difference of extremes of 29.62% and 27.49%. Difference between the two groups



(a) VGG11-BN (b) ResNet50

Figure 5: The retrieval mAP of comparing approaches using VGG11-BN and ResNet50 as the CNN backbone in random camera positions. The proposed *DeepCCFV* had the best performance among the methods listed in the figure. The retrieval mAP of *DeepCCFV* using single view achieved 74.49% and 78.63%, which were 13.69% and 32.02% higher than MVCNN. The retrieval mAP of the proposed *DeepCCFV* using 12 views achieved 82.87% and 87.98% ,which were 6.91% and 11.75% higher than MVCNN.

is even more obvious. *DeepCCFV* and MVCNN with *Dropview* outperformed other two methods with 89.98% and 85.43% result under full-view setting, and they were above the other group all the time.

In this situation, the query views are rendered in arbitrary angle around the 3D object; therefore, it is even harder for MVCNN to generate those strong features which are necessary for retrieval with the given query. Our proposed *DeepCCFV* tackles this issue and provides best retrieval performance compared with other methods listed in the figures. Even if only one or two views are provided in a query, the retrieval performance is adequate compared to the retrieval performance when 12 views are provided.

Cross-modal Retrieval

To further evaluate the effectiveness of our proposed method, we employ point cloud and view based model into a common pipeline for cross-modal retrieval, i.e., the retrieval between 3D objects represented by a single view, 12-views and point cloud. This experiment provides evidence of the robustness of *DeepCCFV* in a complex model; at the same time, it demonstrates the possibility of *DeepCCFV* as a more general model for 3D object representation.

Within the cross-modal pipeline, we first compare the original single-modal retrieval performance of the proposed method DeepCCFV with multi-view based methods MVCNN, GVCNN and point cloud based method DGCNN (Wang et al. 2018). There is no retrieval performance of DGCNN provided. Thus we measure the retrieval mAP of DGCNN by ourselves. Meanwhile, we compare the cross-modal retrieval performance of MVCNN, and the proposed *DeepCCFV* using VGG11-BN as the backbone.

Implementation In our retrieval network, we employ MVCNN which is re-implemented with the same network structure and the proposed method *DeepCCFV*, as the feature extractor of multi-view data. We extract the output of

Method	Query	point cloud	12views	1view	1view	point cloud	12views	1view
	Database	point cloud	12views	1view	12views	12views	point cloud	point cloud
MVCNN (Su et al. 2015)		-	70.10%	61.70%	-	-	-	-
MVCNN,metric (Su et al. 2015)		-	80.20%	-	-	-	-	-
GVCNN (Feng et al. 2018)		-	81.30%	-	-	-	-	-
MVCNN[1]		-	85.52%	-	61.91%	-	-	-
DeepCCFV[1]		-	86.92%	-	**79.57%**	-	-	-
DGCNN		81.60%	-	-	-	-	-	-
MVCNN[1]+ DGCNN		88.65%	88.17%	61.69%	36.06%	88.48%	83.74%	36.88%
DeepCCFV[1]+ DGCNN		**89.33%**	**91.35%**	**80.46%**	76.62%	**90.24%**	**88.51%**	**75.20%**

[1] Using VGG11-BN as CNN backbone.
All view-based models are trained with all 12 views of 3D objects.

Table 1: The results of cross-modal retrieval experiment in terms of mAP. Above the middle line, results show the original single-modal retrieval results for different methods. Below the middle line, results show the cross-modal retrieval result. For all cases, the method with DeepCCFV performs the best.

fc_7 layer as the input of the following embedding network. For point cloud data, we employ Dynamic Graph CNN (Wang et al. 2018), a point-cloud-based model for 3D object representation, as the feature extractor. The output of mlp_1 layer of DGCNN is extracted to be the input of the following embedding network. All of these models are pre-trained on the ModelNet40 dataset.

To narrow down the gap between two modalities and map features from both modalities into one clustered embedding space for cross-modal retrieval, we implement Triplet-Center Loss (He et al. 2018) as retrieval loss in our network, which is calculated in a cross-modal way.

We evaluate the performance of our purposed cross-modal retrieval model in both single-modal way and cross-modal way. The retrieval mAPs of the retrieval between 3D objects represented by point cloud, 12-views and single-view are measured in our experiments. We use the Euclidean distance to measure the similarity of 3D objects. For each query, we select top 1000 results to be our retrieval results.

Results The experimental results of comparisons between different methods and different input regarding mAP are demonstrated in Table 1. Compared with the model that uses MVCNN as the feature extractor of multi-view data, our proposed *DeepCCFV* present superior performance in all seven tasks, especially when the queries are single view data.

Classification Performance

We also conduct 3D object classification experiments as a supplementary experiment. Both VGG11-BN and ResNet50 backbones are evaluated. The results are denoted in Table 2. We can see that reducing the number of views has a significant influence on the original MVCNN model, while it has a weaker influence on *DeepCCFV*. This phenomenon implies that *DeepCCFV* works well on the classification task since it is the same over-fitting issue in retrieval task that hinders the performance of classification.

Backbone	Number of views	Accuracy	
		MVCNN	DeepCCFV
VGG11-BN	1	64.28%	**82.11%**
	2	76.91%	**87.84%**
	4	86.96%	**90.72%**
	8	91.68%	**92.15%**
	12	**92.75%**	92.30%
ResNet50	1	48.11%	**70.39%**
	2	62.24%	**79.69%**
	4	76.44%	**86.09%**
	8	87.58%	**90.74%**
	12	91.98%	**92.46%**

Table 2: The comparison of classification accuracy with different numbers of views between MVCNN and *DeepCCFV*.

Conclusion

In this paper, we propose a camera constraint-free multi-view CNN, i.e., *DeepCCFV*, for 3D object retrieval. A feature augmentation method, named *Dropmax*, has been introduced to overcome the over-fitting issue in multi-view based 3D object representation methods coming from the fixed camera settings. To evaluate its performance, we have conducted extensive experiments comparing with traditional multi-view based methods. At the same time, we also conduct a cross-modal retrieval experiment, which evaluates the performance of the proposed method in a complex model. Both of the results demonstrate the proposed method can achieve stable performance in camera constraint-free environments, particularly when the number of views varies significantly.

In future work, it is worthwhile considering generalizing the concept of *DropMax*, which plays a vital role in

DeepCCFV, to more tasks and more databases. Also, cross-modal retrieval for point cloud and multi-view modalities is a poorly investigated aspect that can be further studied.

Acknowledgements

This work was supported by National Key R&D Program of China (Grant No. 2017YFC0113000).

References

Chang, A. X.; Funkhouser, T.; Guibas, L.; Hanrahan, P.; Huang, Q.; Li, Z.; Savarese, S.; Savva, M.; Song, S.; Su, H.; et al. 2015. ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*.

Chen, D.-Y.; Tian, X.-P.; Shen, Y.-T.; and Ouhyoung, M. 2003. On Visual Similarity Based 3D Model Retrieval. In *Computer Graphics Forum*, volume 22, 223–232. Wiley Online Library.

Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; and Li, F.-F. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 248–255.

Feng, Y.; Zhang, Z.; Xibin, Z.; Ji, R.; and Gao, Y. 2018. GVCNN: Group-View Convolutional Neural Networks for 3D Shape Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 264–272.

Gao, Y.; Tang, J.; Hong, R.; Yan, S.; Dai, Q.; Naiyao, Z.; and Chua, T.-S. 2012. Camera Constraint-Free View-Based 3-D Object Retrieval. *IEEE Transactions on Image Processing* 21:2269–2281.

Guo, H.; Wang, J.; Gao, Y.; Li, J.; and Lu, H. 2016. Multi-View 3D Object Retrieval With Deep Embedding Network. *IEEE Transactions on Image Processing* 25(12):5526–5537.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

He, X.; Zhou, Y.; Zhou, Z.; Bai, S.; and Bai, X. 2018. Triplet-Center Loss for Multi-View 3D Object Retrieval. *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ioffe, S., and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*.

Kanezaki, A.; Matsushita, Y.; and Nishida, Y. 2018. RotationNet: Joint Object Categorization and Pose Estimation Using Multiviews from Unsupervised Viewpoints. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Klokov, R., and Lempitsky, V. S. 2017. Escape from Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models. *2017 IEEE International Conference on Computer Vision (ICCV)* 863–872.

Maturana, D., and Scherer, S. 2015. VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 922–928.

Nitish, S.; Geoffrey, H.; Alex, K.; Ilya, S.; and Ruslan, S. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research* 15(1):1929–1958.

Papadakis, P.; Pratikakis, I.; Theoharis, T.; and Perantonis, S. J. 2009. PANORAMA: A 3D Shape Descriptor Based on Panoramic Views for Unsupervised 3D Object Retrieval. *International Journal of Computer Vision* 89:177–192.

Parkhi, O. M.; Vedaldi, A.; and Zisserman, A. 2015. Deep Face Recognition. In *Proceedings of the British Machine Vision Conference (BMVC)*, volume 1, 6.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 77–85.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, 5105–5114.

Schroff, F.; Kalenichenko, D.; and Philbin, J. 2015. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 815–823.

Simonyan, K., and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*.

Song, H. O.; Xiang, Y.; Jegelka, S.; and Savarese, S. 2016. Deep Metric Learning via Lifted Structured Feature Embedding. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4004–4012.

Su, H.; Maji, S.; Kalogerakis, E.; and Learned-Miller, E. G. 2015. Multi-view Convolutional Neural Networks for 3D Shape Recognition. In *2015 IEEE International Conference on Computer Vision (ICCV)*.

Wang, F.; Xiang, X.; Cheng, J.; and Yuille, A. L. 2017. NormFace: L2 Hypersphere Embedding for Face Verification. In *Proceedings of the 25th ACM International Conference on Multimedia*, 1041–1049. ACM.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2018. Dynamic Graph CNN for Learning on Point Clouds. *arXiv preprint arXiv:1801.07829*.

Xie, J.; Fang, Y.; Zhu, F.; and Wong, E. K. 2015. DeepShape: Deep Learned Shape Descriptor for 3D Shape Matching and Retrieval. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1275–1283.

You, H.; Feng, Y.; Ji, R.; and Gao, Y. 2018. PVNet: A Joint Convolutional Network of Point Cloud and Multi-View for 3D Shape Recognition. In *Proceedings of the 26th ACM International Conference on Multimedia*, 1310–1318. ACM.