Learning Relationships for Multi-View 3D Object Recognition

Ze Yang[1] Liwei Wang[1,2]
[1]Key Laboratory of Machine Perception, MOE, School of EECS, Peking University
[2]Center for Data Science, Peking University
yangze@pku.edu.cn wanglw@cis.pku.edu.cn

Abstract

Recognizing 3D object has attracted plenty of attention recently, and view-based methods have achieved best results until now. However, previous view-based methods ignore the region-to-region and view-to-view relationships between different view images, which are crucial for multi-view 3D object representation. To tackle this problem, we propose a Relation Network to effectively connect corresponding regions from different viewpoints, and therefore reinforce the information of individual view image. In addition, the Relation Network exploits the inter-relationships over a group of views, and integrates those views to obtain a discriminative 3D object representation. Systematic experiments conducted on ModelNet dataset demonstrate the effectiveness of our proposed methods for both 3D object recognition and retrieval tasks.

1. Introduction

We humans live in an environment that 3D object is a ubiquitous and rich source of information. Understanding and representing 3D object has been one of the most fundamental problems in the field of computer vision and artificial intelligence, as it becomes increasingly important for a wide range of practical applications (*e.g.* self-driving, autonomous robot). With the success of deep learning [23, 33, 25, 42] and the rapid development of large-scale 3D repositories [6, 46], various approaches have been proposed for 3D object recognition. Overall speaking, those approaches can be divided into three lines of research according to the input modality: view-based approaches [39, 43, 17, 2, 31, 47, 10, 18], voxel-based approaches [46, 24, 27, 31, 35] and pointset-based approaches [30, 32, 20, 38]. The view-based approaches render the 3D object from multiple views and deploy image-based classifiers on individual view image; The voxel-based approaches represent the object as a 3D occupancy grid and analyze the grid using 3D deep networks. The pointset-based approaches represent the object as a collection of unordered points and make predictions based on the point cloud. Among these approaches, the view-based methods have achieved the best performance so far. Comparing to other input modality, the view-based input is relatively low-dimensional, contains more fine-grained details about the 3D object, and is robust to 3D object representation artifacts. Another advantage of view-based methods is that the input views can be easily captured comparing to other methods, especially for the situation that one cannot directly access to 3D object models. The great success of view-base methods also benefits from the well-established 2D deep representation models, *e.g.*, VGG [37], GoogLeNet [40], ResNet [13] and DenseNet [16].

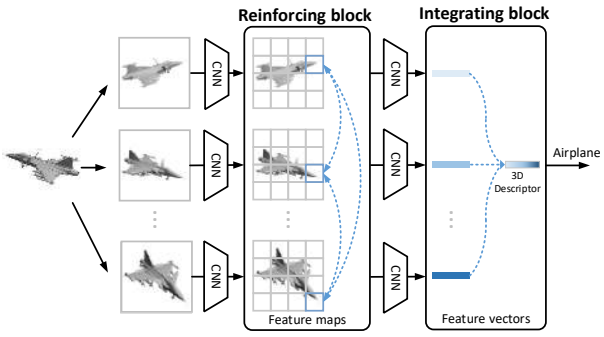

Figure 1. The overview of our method. The Reinforcing block models the region-to-region relationships between different views to reinforce their information. The Integrating block models the view-to-view relationships over views to integrate them into an effective 3D object descriptor.

Intuitively, when humans recognize the 3D object in the world from one particular viewpoint, the object usually have some regions that are occluded, reflective, incomplete or totally invisible to human eyes. Thus humans will need information from other viewpoints to fully understand the object. For this reason, how to combine the information from multiple viewpoints leaves a very important question. Unfortunately, current prevailing view-based methods, in general, cannot solve this problem fundamentally: they lack a mechanism to reason about the relationships of the 2D appearances from different viewpoints. For example, the Multi-View CNN (MVCNN) [39] method uses a view-wise

pooling strategy to fuse the feature from individual views into a global 3D object descriptor. But for a particular part of the 3D object, it will be projected to different spatial positions in the 2D plane if the 3D shape is rendered from different viewpoints. The view-pooling operation ignores the spatial correlation between 2D appearances from different viewpoints, therefore, the corresponding regions from different views are misaligned, and the information from different views cannot be effectively combined. For instance, as shown in Figure 2, the front (nose) of the plane in different views cannot be effectively aggregated through view-pooling. Moreover, view-pooling strategy treats all views equally, without considering the relationships between different views and the discriminative power of each view.

To tackle these issues, we argue that there are two key ingredients: First, modeling the region-to-region relationships. In case that some parts of the 3D object are not clearly understood from some particular viewpoint (*e.g.*, partially occluded, reflective, incomplete), the missing information can be found from other viewpoints. For a given view image, if there is a strategy to matching the regions inside it and the corresponding regions in other views, the information of that given view can be reinforced by exploiting the relationships between matching regions. Second, modeling the view-to-view relationships. In case that several parts of the 3D object are totally invisible from some viewpoints, just modeling region-to-region relationships cannot further help those views to gain information about the invisible parts. We will need to model the view-to-view relationships to determine the discriminative power of each view, and further integrate those views to obtain the final 3D object descriptor.

Motivated by the above analysis, we propose a novel framework for multi-view 3D object recognition. Figure 1 shows an overview of our framework. The model we develop, which we term Relation Network, employs two basic building blocks, *i.e.*, the Reinforcing block and the Integrating block. The Reinforcing block is responsible for exploring region-to-region relationships in order to reinforce the information for each individual view; The Integrating block is responsible for modeling view-to-view relationships so that the information from individual view can be integrated effectively. Specifically, for the feature map of a given view image, each spatial position in the feature map is a feature vector corresponds to a certain region in the image. For each region in the given view image, the Reinforcing block finds the matching/related regions from other views and reinforces the information of that region by taking advantage of the clues from the matching regions. In this way, the information of views can be reinforced. After that, the Integrating block employs a self-attentive selection mechanism to generate the importance score for each view, which denotes the relative discriminative power of that view. Note

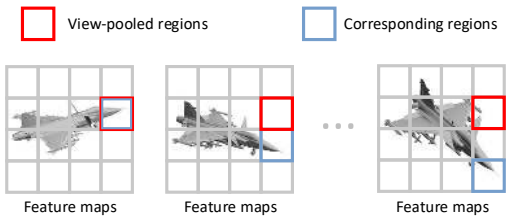

Figure 2. The information from corresponding regions (the front of the plane) should be aggregated, but the view-pooling fuses the information from red boxes.

that to determine whether a certain view is discriminative, we also need to look at the remaining views. This motivates us to consider the relationships over views in the self-attention mechanism. Finally, the Integrating block takes the importance scores and high-level visual features into consideration and generates the final 3D object descriptor.

To demonstrate the effectiveness of the Relation Network, we have conducted systematic experiments on the ModelNet dataset for both 3D object classification and retrieval tasks. Our method uses only greyscale input and achieves state-of-the-art performance. Our main contributions can be summarized as follows:

- Different from the previous multi-view 3D object recognition methods, we utilize a novel relation-based framework to specifically model both the region-to-region relationships and view-to-view relationships over the multi-view input data.
- We propose a Relation Network for the task of 3D object recognition and retrieval. The model contains several Reinforcing and Integrating blocks. The Reinforcing block reinforces the information for individual view by modeling the relationships between its inside regions and the corresponding regions in other views. The Integrating block models the inter-relationships over views and integrates the information from those views to generate a final 3D object descriptor.
- Our model can be trained end-to-end, and we demonstrate the effectiveness of our model on the ModelNet dataset. Our proposed method outperforms existing models in both 3D object recognition and retrieval tasks. Besides, we visualize the behavior of the Reinforcing block and the Integrating block to gain more insights about the framework.

2. Related Works

2.1. Handcraft Descriptors

A growing body literature on handcraft 3D object descriptor analysis has been developed in the field of computer vision and graphics, which can be broadly divided into two categories: shape-based handcraft feature and view-based handcraft feature. shape-based handcraft feature is directly

extracted from the native models. For example, it can be represented as features constructed from curvatures, normals, distances, angles, tetrahedra volumes, triangle areas or local shape diameters of the 3D object surface [29, 8, 14]; mathematical characteristics of spherical functions defined in 3D voxel grids [19]; heat kernel signatures on 3D polygon meshes [5, 22]; or extended SIFT and SURF feature descriptors of 3D volumetric grids [21], *etc*. View-based handcraft feature is extracted from a collection of 2D projections from the 3D object. For instance, the Lighting Field descriptor [9] consists of a set of geometric and Fourier descriptors which are extracted from the object silhouettes. Murase and Nayar [28] compress the set of 2D views to a low-dimensional parametric eigenspace, then they recognize the identity of the 3D object based on the manifold it lies.

2.2. Voxel-based Approaches

Recently, deep neural networks have been applied to 3D object recognition by representing the objects as voxel grids. Wu *et al*. [46] represent the 3D object as a binary 3D tensor, where each voxel can be categorized as free space or occupied space. And they propose the 3D ShapeNets to model the probability distribution of the binary variables in the tensor. Similarly, Maturana and Scherer [27] propose a VoxNet to effectively recognize the 3D object based on the voxel grid. Qi *et al*. [31] propose two distinct volumetric CNN network for 3D object recognition. The first architecture utilizes multi-task training to help the network to scrutinize details of 3D object carefully. The second architecture employs long anisotropic kernels to model the long-distance interactions in the voxel grid. Brock *et al*. [4] propose a Voxception-ResNet (VRN) and achieve a significant improvement. However, these methods are limited by the data sparsity and costly computation of voxel data.

2.3. Pointset-based Approaches

Point cloud is another type of 3D data structure. In the basic setting, a point cloud is a set of unordered points, and each point is represented by its three coordinates in the geometric space plus additional information such as color, normal *etc*. The seminal work of Qi *et al*. [30] uses a PointNet architecture that directly takes unordered point sets as input and outputs 3D object recognition and part segmentation results. However, the PointNet architecture cannot model fine-grained patterns due to its limitation in capturing local structures induced by the metric space points live in. Qi *et al*. [32] later solve this problem and propose a PointNet++ architecture to learn hierarchical features with respect to the distance metric. In parallel, Klokov and Lempitsky [20] propose the Kd-Networks for 3D object recognition by performing multiplicative transformations and using Kd-tree to build the computational graphs.

2.4. View-based Approaches

An alternative direction is to represent the 3D object as a collection of 2D view images. Su *et al*. [39] propose a multi-view convolutional neural network (MVCNN) that first extracts convolutional feature from each view individually, then MVCNN uses a view-pooling strategy to aggregate them into the global 3D representation. Qi *et al*. [31] introduce a new multi-resolution component into the MVCNN, and improve the performance accordingly. Johns *et al*. [17] decompose the image sequence into a collection of image pairs, and then classify each pair independently, finally they learn the 3D object classifier by weighting the contribution of individual pairs. Wang *et al*. [43] propose a view clustering and pooling layer and insert it into the existing model to achieve better performance. Feng *et al*. [10] propose a group-view convolutional neural network (GVCNN) to model the hierarchical correlation in multiple views. Yu *et al*. [47] tackle the problem of 3D object recognition from the perspective of patch similarity and propose a multi-view harmonized bilinear network (MHBN) to obtain the 3D object representation.

2.5. Attention Modules

There has been a long line of research that incorporates attention/relation modules into neural networks for the task of natural language processing [1, 11, 41], computer vision [12, 44, 26, 15] and physical system modeling [3, 34, 45]. An early example is the work by Bahdanau *et al*. [1] that integrates the soft attention mechanism into the RNN unit to enable the model to automatically focus on relevant parts of the source sentence when predicting a target word. Vaswani *et al*. [41] propose a new architecture, the Transformer, to replacing the RNN by attention models entirely. Our method is highly related to these works. We extend the self-attention modules to model the region-to-region relationships and view-to-view relationships in the view sequence. We nontrivially bridge self-attention modules to 3D objects recognition.

3. Methods

3.1. Overview

Our approach rests on the assumption that connecting corresponding regions from different views and reasoning about the relationships between them can help the views to better characterize the 3D object. Therefore, for each region in a given view, the initial goal is to localize its corresponding regions from other views. Formally, we define $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N\}$ as the view sequence that describe the 3D object $\mathcal{X}$, where N is the number of views. In order to obtain region-level features, we extract features from convolutional layer instead of from fully-connected layer. Specifically, for each view $\mathbf{X}_i$, its convolutional fea-

ture map $\mathbf{R}_i \in \mathbb{R}^{L\times L\times D_r}$ can be regarded as $L \times L$ feature vectors, each of which is a D_r-dimensional representation corresponding to a region in the view $\mathbf{X}_i$:

$$\mathbf{R}_i = \{\mathbf{r}_{i1}, \mathbf{r}_{i2}, \cdots, \mathbf{r}_{iL^2}\},\ \mathbf{r}_{ij} \in \mathbb{R}^{D_r}. \tag{1}$$

For the region feature $\mathbf{r}_{ij}$ comes from view $\mathbf{X}_i$, the Reinforcing block enumerates all possible regions in the view sequence and employs a pairwise matching function $\mathcal{M}$ to computes a scalar $M_{ij,mn}$ between $\mathbf{r}_{ij}$ and the feature of each enumerated region $\mathbf{r}_{mn}$, representing how related these two regions are:

$$M_{ij,mn} = \mathcal{M}(\mathbf{r}_{ij}, \mathbf{r}_{mn}). \tag{2}$$

Then the region feature $\mathbf{r}_{ij}$ is reinforced by all enumerated regions depending on their matching score with $\mathbf{r}_{ij}$. We denote the reinforced region feature as $\mathbf{r}^*_{ij}$, it takes advantage of the information from corresponding regions in other views. See more details in Section. 3.2. And we denote the reinforced feature map $\mathbf{R}^*_i \in \mathbb{R}^{L\times L\times D_r}$ for view $\mathbf{X}_i$ as:

$$\mathbf{R}^*_i = \{\mathbf{r}^*_{i1}, \mathbf{r}^*_{i2}, \cdots, \mathbf{r}^*_{iL^2}\},\ \mathbf{r}^*_{ij} \in \mathbb{R}^{D_r}. \tag{3}$$

After that, we send the reinforced feature map $\mathbf{R}^*_i$ through the next part of the network and obtain the reinforced feature vector $\mathbf{f}^*_i \in \mathbb{R}^{D_f}$ for view $\mathbf{X}_i$. Note that the feature vector $\mathbf{f}^*_i$ of a single view cannot encapsulate all information about the 3D object, since there may be some parts of the 3D object that are totally hidden from its viewpoint, we cannot acquire the missing information of those parts through region matching. Consequently, to obtain a complete representation for the 3D object, we will need to combine the information from multiple reinforced feature vectors. To achieve such progress, we propose an Integrating block which exploits the inter-relationships over reinforced feature vectors and generate an importance score for each feature vector. Finally, all reinforced feature vectors and their importance scores are jointly considered to generate a single, compact 3D object descriptor.

3.2. Reinforcing Block

For each view image $\mathbf{X}_i$, the Reinforcing block aims to exploit region-to-region relationships between $\mathbf{X}_i$ and other views, which further help to enhance and refine the information of $\mathbf{X}_i$. Recall from Section 3.1 that the feature extractor produces a set of feature vectors $\mathbf{R}_i = \{\mathbf{r}_{i1}, \mathbf{r}_{i2}, \cdots, \mathbf{r}_{iL^2}\}$ for view $\mathbf{X}_i$, where each feature vector $\mathbf{r}_{ij}$ is a D_r-dimensional representation corresponding to a part region in view $\mathbf{X}_i$. And the matching function $\mathcal{M}$ can compute the matching score $\mathcal{M}_{ij,mn}$ between any two region features $\mathbf{r}_{ij}$ and $\mathbf{r}_{mn}$. Here we choose the embedded dot-product function as the matching function. In this case, we first linearly embed the inputs $\mathbf{r}_1, \mathbf{r}_2 \in \mathbb{R}^{D_r}$ into a hidden space with learnable embedding matrices $W_r \in \mathbb{R}^{D_e\times D_r}$, and we omit the bias term for simplicity. Then we compute their dot-product similarity as the matching score:

$$\mathcal{M}(\mathbf{r}_1, \mathbf{r}_2) = \langle W_r\mathbf{r}_1, W_r\mathbf{r}_2\rangle. \tag{4}$$

After obtaining the matching score between different regions, we denote the matching matrix for $\mathbf{r}_{ij}$ as:

$$\mathbf{M}_{ij} = \begin{bmatrix} M_{ij,11} & M_{ij,12} & \cdots & M_{ij,1L^2} \\ M_{ij,21} & M_{ij,22} & \cdots & M_{ij,2L^2} \\ \vdots & \vdots & \ddots & \vdots \\ M_{ij,N1} & M_{ij,N2} & \cdots & M_{ij,NL^2} \end{bmatrix}. \tag{5}$$

Due to the low-dimensionality D_e of the embedded hidden space, the matching matrix can be computed efficiently using highly optimized matrix multiplication code. Intuitively, the regions with higher matching scores with $\mathbf{r}_{ij}$ should be more likely to represent the same part of the 3D object, therefore are more likely to augment the information of $\mathbf{r}_{ij}$. We first normalize the matching matrix $\mathbf{M}_{ij}$ and then compute the reinforced region feature $\mathbf{r}^*_{ij}$ as follows:

$$\hat{\mathbf{M}}_{ij} = \text{Normalize}(\mathbf{M}_{ij}), \tag{6}$$

$$\mathbf{r}^*_{ij} = \mathbf{r}_{ij} + f\left(\sum_{m=1}^{N}\sum_{n=1}^{L^2} \hat{M}_{ij,mn}\cdot g(\mathbf{r}_{mn})\right), \tag{7}$$

where we interpret g as a projection that maps the region feature into a space with some nice properties (*e.g.*, the information of region features can be easily fused). The information of $\mathbf{r}_{ij}$ is reinforced by computing the weighted sum (in projected space) of all enumerated region features, and the weight for each region feature is its normalized matching score with $\mathbf{r}_{ij}$. Function f is a mapping that injects the weighted sum into original region feature space. For simplicity, we consider the situation that f and g in the form of linear mappings, *i.e.*, $g(\mathbf{x}) = W_g\mathbf{x}$ and $f(\mathbf{x}) = W_f\mathbf{x}$, where $W_g \in \mathbb{R}^{D_g\times D_r}$ and $W_f \in \mathbb{R}^{D_r\times D_g}$ are learned parameters, we omit bias term for simplicity. The Reinforcing block is smooth and differentiable, making the network end-to-end trainable. Note that we use a residual connection in the Reinforcing block. The residual connection makes the network easier to optimize, the Reinforcing block can start as the identity mapping and gradually transform to be more task-oriented. Next we describe how to normalize the matching matrix.

Normalizing across views. In this setting, the matching matrix is normalized over all the entries using scaled softmax function:

$$\hat{M}_{ij,mn} = \frac{e^{\frac{M_{ij,mn}}{\sqrt{D_e}}}}{\sum_{m=1}^{N}\sum_{n=1}^{L^2} e^{\frac{M_{ij,mn}}{\sqrt{D_e}}}}, \tag{8}$$

where D_e is the row size of W_r. However, this strategy may lead to the problem of dominating phenomenon, *i.e.*, the matching matrix is dominated by very few entries. In our task, for a given region in a reference view, we hope to find sufficient regions in other views that can match the given region.

Normalizing inside Views. In this setting, each row of the matching matrix is normalized independently:

$$\hat{M}_{ij,mn} = \frac{e^{\frac{M_{ij,mn}}{\sqrt{D_e}}}}{N \cdot \sum_{n=1}^{L^2} e^{\frac{M_{ij,mn}}{\sqrt{D_e}}}}. \tag{9}$$

Under such situation, the matching score is normalized over regions inside each view. More concretely, for one given region in a reference view, this approach selects the best matching regions from each view independently, and then combines all the matching regions to reinforce the information of the given region. We employ this normalization strategy in the Reinforcing block.

3.3. Integrating Block

Given the reinforced feature vectors for all views, the objective is to integrate them and to generate the final representation for the 3D object. In the simplest setting, the 3D object feature can be represented via view pooling operation over all the reinforced feature vectors. However, the relationships between different views and the discriminative power of each view are ignored in this setting.

To overcome these issues, we propose a generic Integrating block to model the inter-relationships between different views, and assigns each view with an importance score. The final 3D object descriptor is derived based on all reinforced feature vectors and their importance score. Intuitively, to determine whether a certain view is discriminative, we also need to look at the remaining views. Consequently, it is suboptimal to use a unary function to compute the importance score for each view. In other words, for each reinforced feature vector $\mathbf{f}_i^* \in \mathbb{R}^{D_f}$, its importance score should be computed based on the relationship between $\mathbf{f}_i^*$ and all other reinforced feature vectors. Formally, we define the importance function $\mathcal{I}$ to computes the importance score I_i for each reinforced feature vector $\mathbf{f}_i^*$ in the form of:

$$\begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} = \mathcal{I}(\mathbf{f}_1^*, \mathbf{f}_2^*, \cdots, \mathbf{f}_N^*). \tag{10}$$

Note that the importance function $\mathcal{I}$ should support orderless inputs and variable number of inputs, and it needs to have fewer parameters. We design the importance function $\mathcal{I}$ as the combination of pair-wise function as in the following:

$$\mathcal{I}(\mathbf{f}_1^*, \mathbf{f}_2^*, \cdots, \mathbf{f}_N^*) = \sum_{j=1}^{N} \begin{bmatrix} \mathcal{R}(\mathbf{f}_1^*, \mathbf{f}_j^*) \\ \mathcal{R}(\mathbf{f}_2^*, \mathbf{f}_j^*) \\ \vdots \\ \mathcal{R}(\mathbf{f}_N^*, \mathbf{f}_j^*) \end{bmatrix}, \tag{11}$$

where $\mathcal{R}$ is a scalar pairwise function. Without loss of generality we use the embedded dot-product function $\mathcal{R}(f_1^*, f_2^*) = \langle W_a \mathbf{f}_1^*, W_a \mathbf{f}_2^* \rangle$, where $W_a \in \mathbb{R}^{D_a \times D_f}$ is learnable, we omit bias term for simplicity. In this setting, the importance score I_i for reinforced feature vector $\mathbf{f}_i^*$ is computed as:

$$I_i = \sum_{j=1}^{N} \mathcal{R}(\mathbf{f}_i^*, \mathbf{f}_j^*). \tag{12}$$

We normalize the importance scores using ReLU normalization as in Equation 13. Note that using softmax function here may lead to the problem of gradient saturation, and therefore making the training unstable. Then the output $\mathbf{f}$ is calculated as the convex combination of the reinforced feature vectors:

$$\hat{I}_i = \frac{\text{ReLU}(I_i)}{\sum_{j=1}^{N} \text{ReLU}(I_j)}, \tag{13}$$

$$\mathbf{f} = \sum_{i=1}^{N} \hat{I}_i \cdot \mathbf{f}_i^*. \tag{14}$$

We then send $\mathbf{f}$ through the remaining fully-connected layers to obtain the final 3D object representation.

3.4. Relation Network

It is straightforward to apply the Reinforcing block and Integrating block to existing architecture. To make a fair comparison with previous approaches, we use the VGG-M network as the base model. By inserting the Reinforcing block and Integrating block into the architecture, we construct a Relation Network. In default, the Reinforcing block is inserted after the conv5 layer, and the Integrating block is inserted after the fc6 layer. Noting that the building blocks can be placed in different positions in the network. Following the similar schemes, more variants that integrate with other base architecture can be constructed.

3.5. Discussion

It is interesting to observe that our formulation combined with Eqn. 8 is very similar to the recently proposed Non-local Network [44] and Transformer [41]. However, compared with these existing works, our Reinforcing block and Integrating block enjoy several unique advantages for 3D object recognition: 1) the Non-local block treats the

spatial and temporal relations with no differentiation. Unlike video, the object exists in every view. In our solution (Eqn. 9), for one given region in a reference view, the Reinforcing block selects the best matching regions from each view independently, this strategy encourages the given region to gain information from all other views. 2) In our Integrating block, we normalize the importance scores using ReLU function (Eqn. 13), it can be seen as a first-order approximation of the Softmax function, which yields better performance and more stable training. To the best of our knowledge, we are the first to bridge self-attention models to multi-view 3D objects recognition, and to solve the problem of region misalignment between different views in 3D object recognition.

4. Experiments

We evaluate our proposed Relation Network and discuss the results with current state-of-the-art methods on the benchmark ModelNet [46] in Section 4.3. Then, we study the influence of the number of views on the performance in Section 4.4. Next, we investigate the influence of the building block in Section 4.5. Finally, we visualize the behavior of Reinforcing block and Integrating block in Section 4.6.

4.1. Dataset

We use ModelNet dataset [46] to evaluate the performance of the Relation Network. ModelNet currently contains 127,915 3D CAD models from 662 categories. The models are collected by using the online search engine and annotated by the workers on Amazon Mechanical Turk. ModelNet40 is a subset including 12,311 models from 40 categories, and the models are split into 9,843 training examples and 2,468 testing examples. In addition, ModelNet10 is another subset consists of 4,899 models from 10 categories, the models are split into 3,991 training examples and 908 testing examples. Both ModelNet40 and ModelNet10 are well annotated and can be downloaded online. Note that the numbers of examples are not equal across different classes, hence we report both average instance accuracy and average class accuracy following previous works. Average instance accuracy counts the percentage of correctly predicted examples among all the examples, while average class accuracy is the average of each accuracy per class (sum of accuracy for each class / total number of class).

4.2. Implementation Details

In all our experiments, We use VGG-M [7] pre-trained on ImageNet as the base model. This allows us to make a fair comparison with existing methods which are mostly based on VGG-M. We use SGD optimizer with learning rate 0.001, momentum 0.9, and we use random horizontal view flipping and weight decay 0.0001 to reduce overfitting. For multi-view input generation, we assume the shapes are upright oriented along a specific axis (*e.g.*, the z-axis), and we set the virtual cameras to point towards the centroid of the mesh, elevated by 30 degrees from the ground plane. The viewpoints are circled placed at intervals of a given angle θ around the axis. We set $\theta = 30$ in default, which generates 12 views for each 3D object. And we study the influence of θ (number of views) in Section 4.4.

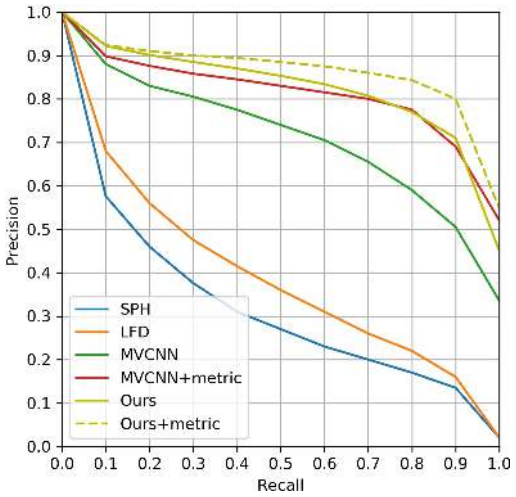

Figure 3. The precision-recall curves for 3D object retrieval on the ModelNet40 dataset. Our Relation Network with metric learning achieves the best performance of 86.7 mAP.

4.3. 3D Object Classification and Retrieval

We first compare our method with those using handcraft descriptors, including SPH [19] and LFD [9]. Then we compare with the voxel-based approaches, including 3D ShapeNets [46], VoxNet [27], Subvolume Supervision Network [31], and Voxception-ResNet [4]. Next, we compare with the voxel-based approaches, including PointNet [30], PointNet++ [32], and Kd-Networks [20]. Finally, our method is compared with view-based approaches, including MVCNN [39], MVCNN-MultiRes [31], Pairwise Decomposition Network [17], RCPCNN [43], GVCNN [10], and MHBN [47].

Table 1 shows the experimental results and comparisons for 3D object recognition and retrieval. Our proposed method achieves the best performance of $94.3\%/92.3\%$ average instance/class accuracy on ModelNet40 dataset and $95.3\%/95.1\%$ average instance/class accuracy on ModelNet10 dataset, demonstrating the effectiveness of our method. Note that the Relation Network uses single-modality input, while some other approaches use multi-modality input and more advanced deep networks. The MHBN [47] is a strong competitor, which achieves 94.1% average instance accuracy on ModelNet40 dataset. However, the MHBN is sensitive to the number of views, we observe a performance drop when the number of views increases from 6 to 12. Also, the MHBN requires computing

Methods	Input Modality	ModelNet40		ModelNet10		Retrieval on ModelNet40
		Inst Acc	Class Acc	Inst Acc	Class Acc	
SPH [19]	Handcraft	-	68.2	-	-	33.3
LFD [9]	Handcraft	-	75.5	-	-	40.9
3D ShapeNets [46]	Volume	-	77.3	-	83.5	49.2
VoxNet [27]	Volume	-	83.0	-	92.0	-
Subvolume Net [31]	Volume	89.2	86.0	-	-	-
Voxception-ResNet [4]	Volume	91.3	-	93.6	-	-
PointNet [30]	Points	89.2	86.2	-	-	-
PointNet++ [32]	Points w/ Normal	91.9	-	-	-	-
Kd-Networks [20]	Points	91.8	88.5	94.0	93.5	-
MVCNN [39]	12 Views	92.1	89.9	-	-	80.2
MVCNN-MultiRes [31]	Multi-resolution Views	93.8	91.4	-	-	-
Pairwise Network [17]	12 Views w/ Depth	-	91.1	-	93.2	-
RCPCNN [43]	12 Views w/ Depth, Normal	93.8	-	-	-	-
GVCNN [10]	8 Views	93.1	-	-	-	84.5
	12 Views	92.6	-	-	-	85.7
MHBN [47]	6 Views	94.1	92.2	94.9	94.9	-
	12 Views	93.4	-	-	-	
Ours	12 Views	**94.3**	**92.3**	**95.3**	**95.1**	**86.7**

Table 1. Comparisons of performance with state-of-the-art methods. Numbers are reported in percentage. Our Relation Network achieves the best performance consistently.

bilinear features and singular value decomposition (SVD), which is tedious and computational costly in practice. In the 3D object retrieval task, our Relation Network achieves the best retrieval performance with 82.7 mAP. However, the Relation Network feature is trained directly for classification, and thus not optimized for retrieval. As suggested in [39], we further adopt a low-rank Mahalanobis metric W that directly projects 3D object feature into a 128-dimensional space, such that the intra-class distance in the projected space is smaller and inter-class distance is larger. Similar to previous works, we adopt the large-margin metric learning algorithm and implementation from [36]. By learning the low-rank Mahalanobis metric, our Relation Network further achieves the best performance of 86.7 retrieval mAP. The precision-recall curves are shown in Figure 3.

4.4. Influence of the Number of Views

To investigate the influence of the number of views on the classification performance, we vary the view number for training and testing. We compare it with the MVCNN [39], RCPCNN [44], GVCNN [10] and MHBN [47]. The accuracies of the compared methods are taken from Yu *et al*. [47] and Wang *et al*. [43]. The average instance accuracy on the ModelNet40 dataset are provided in Table 2. It can be observed that our Relation Network outperforms or achieves comparable result with previous methods. Table 2 shows that previous methods suffer from a performance drop when the number of views increases from 6 to 12. However, our Relation Network benefits from the increase of view number coherently, demonstrating that the Relation Network is able to model the relationships between views robustly.

Methods	Number of Views		
	3 views	6 views	12 views
MVCNN [39]	91.3	92.0	91.5
RCPCNN [43]	92.1	92.2	92.2
MHBN [47]	**93.8**	**94.1**	93.4
Ours	93.5	**94.1**	**94.3**

Table 2. Experiments on ModelNet40 when the number of views for training and testing is varying. Average instance accuracies are reported in percentage. The accuracies of MVCNN, RCPCNN and MHBN are taken from [43] and [47].

4.5. Influence of the building block

In order to analyze the influence of different components in our Relation Network, we design different running settings on ModelNet40 to study the positions and numbers of building blocks. Note that the VGG-M network is a relatively shallow network, consisting of only 5 convolutional layers and 2 fully-connected layers. So the Integrating block is placed between 2 fully-connected layers. We study the positions and numbers of our building blocks. Table 3 compares a single Reinforcing block added to different positions of the VGG-M, and we found that placing the Reinforcing block after conv5 achieves slightly better perfor-

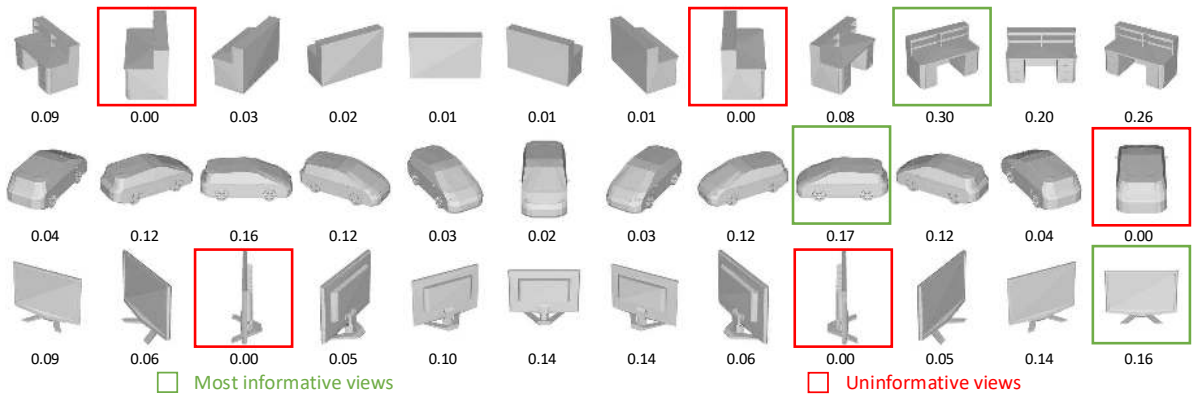

Figure 4. Visualization of the importance scores computed by the Integrating block. The weight of each view is shown at the bottom of the image. The most informative regions are framed with green rectangles, and the least informative regions are framed with red rectangles.

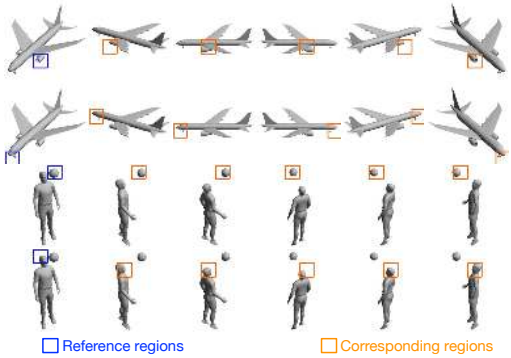

Figure 5. Visualization of the corresponding regions computed by the Reinforcing block placed after conv5. Each row provides an example of region. For each reference region, its most relevant regions in other views are framed with orange rectangles.

mance, we hypothesis it is because the conv5 has a larger receptive field, which eases region matching between different views. As shown in Table 3, when we replace the Integrating block to view pooling strategy, the performance decreases from 94.3 to 93.8. Table 4 shows the results of placing multiple Reinforcing blocks into the network. We found that if no Reinforcing block is inserted, the performance will slightly drop, and placing more Reinforcing blocks than one does not improve the performance.

w/ Integrating block	Places of a single Reinforcing block	Average Instance Accuracy
✓	conv2	93.9
✓	conv3	93.7
✓	conv4	94.0
✓	conv5	**94.3**
×	conv5	93.8

Table 3. Study of the influence of adding a single Reinforcing block to different positions.

Places of multiple Reinforcing blocks	Average Instance Accuracy
w/o Reinforcing block	93.7
conv5	**94.3**
conv4,5	93.9
conv3,4,5	94.2
conv2,3,4,5	93.8

Table 4. Study of the influence of adding multiple Reinforcing blocks.

4.6. Visualization of Attention

We further select some 3D models from testing set and visualize the behavior of Reinforcing block and Integrating block in Figure 5 and Figure 4. For a given reference region, we find its most corresponding bins computed by the Reinforcing block after conv5 from the feature map of other views, and draw orange rectangles in Figure 5. We found that the Reinforcing block focuses well on the corresponding regions from other views. Figure 4 shows the importance scores of different views computed by the Integrating block. We found that views contain more discriminative parts of the 3D object will be assigned higher importance scores. Then learned attention is meaningful and consistent with human intuition.

5. Conclusions

In this paper, we propose a novel Relation Network for 3D object recognition and retrieval. The region-to-region relationships and view-to-view relationships are both taken into considerations. The Relation Network is end-to-end trainable and achieves the state-of-the-art performance on ModelNet dataset. To gain more understanding about our framework, we have conducted systematic experiments to investigate the effects of different components in our method. And we show the learned attention corresponds to human intuition well.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015. 3

[2] Song Bai, Xiang Bai, Zhichao Zhou, Zhaoxiang Zhang, and Longin Jan Latecki. Gift: A real-time and scalable 3d shape search engine. In *CVPR*, pages 5023–5032, 2016. 1

[3] Peter W Battaglia, Razvan Pascanu, Matthew Lai, Danilo Rezende, and Koray Kavukcuoglu. Interaction networks for learning about objects, relations and physics. In *NIPS*, 2016. 3

[4] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. In *NIPS*, 2016. 3, 6, 7

[5] Alexander M Bronstein, Michael M Bronstein, Leonidas J Guibas, and Maks Ovsjanikov. Shape google:geometric words and expressions for invariant shape retrieval. *Acm Transactions on Graphics*, 30(1):1–20, 2011. 3

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1

[7] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014. 6

[8] Siddhartha Chaudhuri and Vladlen Koltun. Data-driven suggestions for creativity support in 3d modeling. In *ACM SIGGRAPH Asia*, page 183, 2010. 3

[9] Ding-Yun Chen, Xiao-Pei Tian, Yu-Te Shen, and Ming Ouhyoung. On visual similarity based 3d model retrieval. In *Computer Graphics Forum*, pages 223–232, 2003. 3, 6, 7

[10] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *CVPR*, 2018. 1, 3, 6, 7

[11] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *ICML*, 2017. 3

[12] Rohit Girdhar and Deva Ramanan. Attentional pooling for action recognition. In *NIPS*, 2017. 3

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 1

[14] Berthold Klaus Paul Horn. Extended gaussian images. *Proceedings of the IEEE*, 72(12):1671–1686, 1984. 3

[15] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *CVPR*, 2018. 3

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, pages 4700–4708, 2017. 1

[17] Edward Johns, Stefan Leutenegger, and Andrew J. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *CVPR*, pages 3813–3822, 2016. 1, 3, 6, 7

[18] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *CVPR*, 2018. 1

[19] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3d shape descriptors. *Proc. Symposium of Geometry Processing*, 43(2):156–164, 2003. 3, 6, 7

[20] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *CVPR*, 2017. 1, 3, 6, 7

[21] Jan Knopp, Mukta Prasad, Geert Willems, Radu Timofte, and Luc Van Gool. Hough transform and 3d surf for robust three dimensional classification. In *ECCV*, pages 589–602, 2010. 3

[22] Iasonas Kokkinos, Michael M Bronstein, Roee Litman, and Alex M Bronstein. Intrinsic shape context descriptors for deformable shapes. In *CVPR*, pages 159–166, 2012. 3

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 1

[24] Yangyan Li, Sören Pirk, Hao Su, Charles R Qi, and Leonidas J Guibas. Fpnn: Field probing neural networks for 3d data. In *NIPS*, 2016. 1

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[26] Chih-Yao Ma, Asim Kadav, Iain Melvin, Zsolt Kira, Ghassan AlRegib, and Hans Peter Graf. Attend and interact: Higher-order object interactions for video understanding. In *CVPR*, 2018. 3

[27] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, pages 922–928, 2015. 1, 3, 6, 7

[28] Hiroshi Murase and Shree K Nayar. Visual learning and recognition of 3-d objects from appearance. *International journal of computer vision*, 14(1):5–24, 1995. 3

[29] Robert Osada, Thomas Funkhouser, Bernard Chazelle, and David Dobkin. Shape distributions. *Acm Transactions on Graphics*, 21(4):807–832, 2002. 3

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1, 3, 6, 7

[31] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *CVPR*, pages 5648–5656, 2016. 1, 3, 6, 7

[32] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. 2017. 1, 3, 6, 7

[33] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99, 2015. 1

[34] Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, 2017. 3

[35] Nima Sedaghat, Mohammadreza Zolfaghari, Ehsan Amiri, and Thomas Brox. Orientation-boosted voxel nets for 3d object recognition. In *BMVC*, 2017. 1

[36] Karen Simonyan, Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman. Fisher vector faces in the wild. In *BMVC*, 2013. 7

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[38] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *CVPR*, June 2018. 1

[39] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, pages 945–953, 2015. 1, 3, 6, 7

[40] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 1

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 3, 5

[42] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015. 1

[43] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. In *BMVC*, 2017. 1, 3, 6, 7

[44] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, June 2018. 3, 5, 7

[45] Nicholas Watters, Daniel Zoran, Theophane Weber, Peter Battaglia, Razvan Pascanu, and Andrea Tacchetti. Visual interaction networks. In *NIPS*, 2017. 3

[46] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 1, 3, 6, 7

[47] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *CVPR*, 2018. 1, 3, 6, 7