View-GCN: View-based Graph Convolutional Network for 3D Shape Analysis

Xin Wei, Ruixuan Yu and Jian Sun (✉)
Xi'an Jiaotong University, Xi'an, 710049, China
{wxmath,yuruixuan123}@stu.xjtu.edu.cn,jiansun@xjtu.edu.cn

Abstract

View-based approach that recognizes 3D shape through its projected 2D images has achieved state-of-the-art results for 3D shape recognition. The major challenge for view-based approach is how to aggregate multi-view features to be a global shape descriptor. In this work, we propose a novel view-based Graph Convolutional Neural Network, dubbed as view-GCN, to recognize 3D shape based on graph representation of multiple views in flexible view configurations. We first construct view-graph with multiple views as graph nodes, then design a graph convolutional neural network over view-graph to hierarchically learn discriminative shape descriptor considering relations of multiple views. The view-GCN is a hierarchical network based on local and non-local graph convolution for feature transform, and selective view-sampling for graph coarsening. Extensive experiments on benchmark datasets show that view-GCN achieves state-of-the-art results for 3D shape classification and retrieval.

1. Introduction

3D shape recognition is an important research area in computer vision. 3D shapes, including real scanned or CAD objects, retain richer geometric and shape information for recognition than the 2D images captured from a single view. 3D shape recognition plays a critical role in applications such as automatic drive [40], archaeology [44], virtual reality / augmented reality [17], *etc*.

Tremendous advances on 3D shape analysis have been achieved in recent years. According to 3D shape representation, these methods can be divided into three categories, *i.e*., voxel-based, point-based, and view-based methods. *Voxel-based methods* represent a 3D shape by a collection of voxels in 3D Euclidean space, then build neural networks on voxels to learn the features for recognition [37, 51]. Though they are effective in performance, they commonly have challenges including the computational complexity, voxel resolution, and data sparsity caused by voxelization of the shape surface. *Point-based methods* directly define networks on point cloud or mesh. PointNet [7] is a simple but powerful deep architecture that takes point positions as input. Succeeding methods, *e.g*., PointNet++ [42], SpiderCNN [52], PointCNN [32], RS-CNN [33] achieve improved performance for 3D shape recognition. *View-based methods* [8, 15, 20, 27, 28, 41, 47, 50, 54] are based on aggregation of multi-view features for recognizing shape categories based on multi-view 2D images. They are among the state-of-the-art methods for 3D shape recognition by taking advantages of 2D image classification networks.

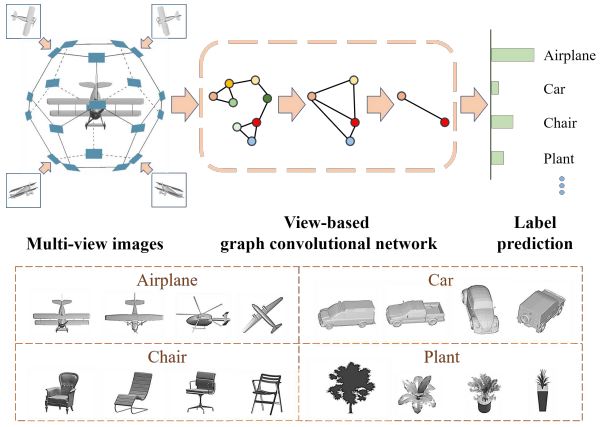


Figure 1. Illustration of view-GCN. Taking multi-view images of a 3D object as a view-graph, view-GCN gradually aggregates multi-view features over view-graph, and outputs class label.

This work focuses on view-based approach. The major challenge is how to aggregate multi-view features to be a global 3D shape descriptor. Traditional methods [47, 48] aggregate multi-view features by max-pooling, which is permutation invariant but ignores relations among views. Taking multiple views as a sequence, RNN has been successfully applied to fuse multi-view features in [8, 20, 34]. View-gram [23] and 3d2seqviews [19] also represent multiple views as a sequence and use convolution and attention to investigate relations of views to learn shape descriptors. The 1D sequential representation can well model the configuration that cameras are sequentially located on a circle around object. However, for general configurations, *e.g*.,

cameras on dodecahedron, 1D sequence ignores 3D geometry of multiple views. Both RotationNet [28] and EMV [13] explore more general view configurations, respectively find best poses by rotations and apply rotation group convolution, however, they rely on assumption of homogeneous space (*e.g.*, icosahedron) for view configurations.

In this work, we present a flexible representation of multiple views of 3D shape by *view-graph*, and each view corresponds to a graph node equipped with view features. The graph edges among nodes are determined by k-nearest neighbor of camera coordinates. We design a novel Graph Convolutional Network (GCN) over view-graph to aggregate multi-view features to learn global shape descriptor, as shown in Fig. 1. The major advantages of this view-graph based representation are as follows. First, it can flexibly model different view configurations, *e.g.*, cameras located on circles, corners of dodecahedron or even irregular positions around objects. Second, by using view-graph representation, we can take advantage of GCN to aggregate multi-view features considering relations of graph nodes.

Along this idea, we propose a novel GCN on view-graph, dubbed as *view-GCN*, to learn 3D shape descriptor. Each node of the view-graph is equipped with extracted features by a backbone 2D image classification network. The proposed view-GCN is a hierarchical GCN architecture with multiple levels over increasingly coarsened view-graphs. In each level, a local graph convolution operation and a non-local message passing operation are designed to aggregate multi-view features by investigating relations among neighboring views and long-range paired views. For graph coarsening, we develop a selective view-sampling strategy to sample representative views by view-selectors. All the learned features in different levels are combined to be a global shape descriptor.

By evaluating on 3D datasets for shape classification and retrieval, view-GCN achieves state-of-the-art performance, *e.g.*, 96.5% per class and 97.6% per instance classification accuracies on ModelNet40 [51], 78.4% micro-averaged and 60.2% macro-averaged retrieval mAP on ShapeNet Core55 [46]. It also outperforms the current state-of-the-art methods on real multi-view dataset RGBD [31].

2. Related Works

2.1. Multi-view 3D shape recognition

View-based approach is effective for 3D shape recognition. MVCNN [47] relies on 2D image classification network to extract multi-view features, then aggregates them by max-pooling to obtain a compact shape descriptor. Several works consider advanced feature aggregation strategies. Both GVCNN [15] and RCPCNN [50] group multi-view features and design feature pooling on view groups. MHBN [55] and RN [53] aggregate features of multi-view patches by harmonized bilinear pooling and attention. RN [53] further models relations over a group of views and integrates them to a shape descriptor using relation scores.

Another interesting strategy is to explore rotations of views. RotationNet [28] treats the view index as latent variable to find best pose when predicting shape label. EMV [13] is based on a discrete subgroup of rotation group and apply a group convolution to homogeneous space of views, *e.g.*, icosahedron. Recently, several works represent multiple views as a sequence. 3D2SeqViews [19] and VNN [23] apply a view-wise convolution on consecutive view sub-sequences in a circular trajectory and then aggregate features by attention. Sequential views are selected and (or) aggregated by RNN, *e.g.*, GRU or LSTM, in VERAM [8], Point2Sequence [20] and Ma *et al*. [34].

Compared with them, we represent multiple views of a 3D shape by a view-graph. The view-graph representation enables us to design GCN to aggregate multi-view features by investigating relations of views. This view-graph representation is more general than sequential representation [19, 23], and view-to-view relations [53].

2.2. Graph convolutional networks

Graph convolutional networks [6, 10, 24, 29] are powerful tools for analyzing graph data. Spectral GCNs [6, 10, 24] represent graph by spectrum of graph Laplacian. They design transforms such as polynomial transform [10] and convolution [6, 24] based on spectral representation. To reduce computational overhead, [10] approximates eigen-decomposition by Chebyshev polynomials.

Recent GCNs [12, 18, 45] conduct spatial convolution by aggregating node features in local neighborhoods on graph, *e.g.*, GCN in [29], GraphSAGE [18]. Graph attention network [49] uses attention to specify different weights to different nodes in neighborhood. Message passing network [16] is based on edge embedding to cumulate messages from neighboring nodes for updating node features. The similar idea is utilized in recurrent relational network [45] for relational reasoning. In [4, 36], local irregular points are quantized in regular grids by angular and radial bins, on which regular 2D convolutions can be defined.

In our work, we represent multiple views of a 3D shape by a view-graph and define GCN on view-graph. This is a novel application of GCN. Our view-GCN is inspired by current GCNs but is carefully designed to hierarchically aggregate multi-view features with local and non-local operations, and a novel selective view-sampling strategy for graph coarsening. Experiments and ablation study justify effectiveness of our view-GCN for 3D shape recognition.

3. Overview of Our Approach

We first discuss on the motivation and overview of our view-GCN for 3D shape descriptor learning.

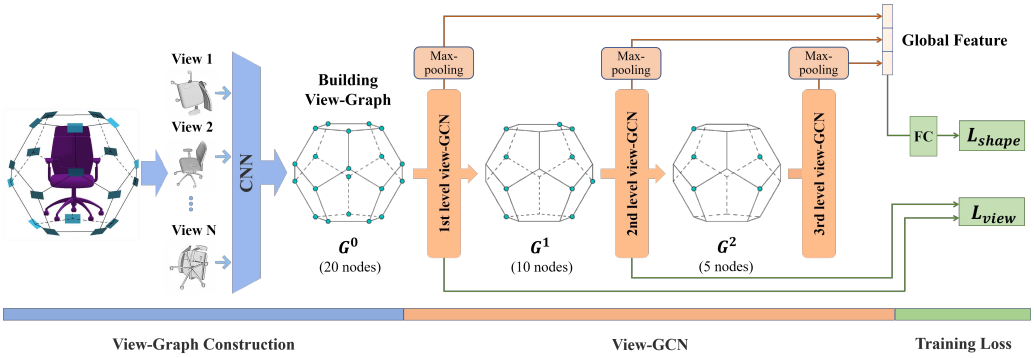
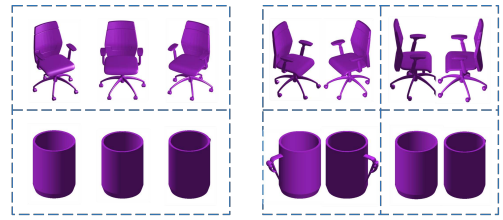


Figure 2. Overview of our approach. It consists of three components, *i.e.*, view-graph construction, view-based graph convolutional network (view-GCN) and traning loss. View-GCN is a hierarchical network defined on gradually coarsened view-graphs.



(a) 2D images in neighboring views. (b) 2D images in symmetric views.

Figure 3. Neighboring and symmetric views of 3D objects.

3.1. Motivation

As shown in Fig. 3 (a), neighboring views of a chair are smoothly changed in pose and appearance, while multiple views of a cup are almost the same. This provides discriminative information for recognition. Moreover, paired views are also related, *e.g.*, the paired views are symmetric in Fig. 3 (b). These relations among multi-view images encode latent geometry of objects, providing valuable information for recognizing 3D objects. It is challenging to enumerate and model all possible relations among views, but this phenomenon inspires us to design a graph convolutional network to automatically investigate relations among views when aggregating multi-view features.

3.2. General pipeline

We design a novel *view-based Graph Convolutional Network*, dubbed as *view-GCN*, to hierarchically aggregate multi-view features considering relations of views by graph convolution. As illustrated in Fig. 2, our method consists of three components. First, multi-view features are extracted by backbone network from multiple views of a 3D object, and then we build a view-graph with nodes represented by view features. Second, we design a GCN to hierarchically aggregate multi-view features on view-graph to generate a global shape descriptor. Finally, the global shape descriptors are taken for shape recognition.

4. From 3D Shape to View-Graph

We now introduce how to construct a view-graph for a 3D shape. We build a directed graph G with i-th node as the i-th view with camera coordinates v_i. This graph is termed as *view-graph*, then adjacency matrix $S \in R^{N\times N}$ of the view-graph is

$$S_{ij} = \Phi(g_{ij}; \theta_s) \tag{1}$$

where $g_{ij} = [v_i, v_j, v_i - v_j, \|v_i - v_j\|_2] \in R^{10}$ represents the spatial relation of two views and $[\,]$ denotes vectorized concatenation of elements. Φ is a non-linear embedding with parameters θ_s for paired nodes. In implementation, we set Φ as a three-layer MLP with LeakyReLU and 10 hidden units in first two layers and it outputs a scalar S_{ij}.

We further use k-nearest neighbor (kNN) to find fixed number of neighboring nodes of each node using coordinate distance, and only keep edges between neighboring nodes. Therefore element of sparse adjacency matrix A is

$$A_{ij} = S_{ij} \cdot \mathbb{I}\{v_j \in \mathcal{N}(v_i)\} \tag{2}$$

where $\mathbb{I}(\cdot)$ is a binary function indicating whether v_j is within the kNN of v_i. Obviously, view-graph can represent different view configurations, *e.g.*, the circular, dodecahedral, irregular configurations as shown in Fig. 4, in which each 3D rectangle represents a view.

5. View-based Graph Convolutional Network

View-GCN is a hierarchical structure defined on multiple levels of coarsened view-graphs. Initially, the view-graph G^0 is defined over all input views, and each view is with an extracted view feature vector as discussed in Sect. 5.1. As shown in Fig. 5, at l-th level, the view-graph G^l is with N_l nodes, *i.e.*, views. Over the view-graph, we successively update the node features by *local graph convolution* and *non-local message passing*. Then the graph G^l is coarsened by

(a) (b) (c)

Figure 4. Different view configurations. Sub-figures respectively correspond to circular, dodecahedral and irregular configurations.

our proposed *selective view-sampling strategy* to construct next level view-graph G^{l+1} for increasing receptive field to benefit semantic feature learning. Features of all levels are fused to be a global shape descriptor.

Compared with max-pooling that merges all multi-view features in a single pooling operation, our view-GCN gradually merges multi-view features on hierarchically coarsened view-graphs considering relations of views, and all features in all levels are utilized in shape descriptor.

5.1. Initial view feature extraction

Given N views $\{I_i\}_{i=1}^N$, features $\{\boldsymbol{f}_i^0\}_{i=1}^N$ are extracted by a fine-tuned 2D image classification network, *e.g.*, ResNet-18 [21] pre-trained on ImageNet [11]. The network is fine-tuned on shuffled multi-view 2D images of all training 3D objects for classification, and features of different views before last FC (fully connected) layer are vectorized to be view features, as initializations of node features in G^0.

5.2. Local graph convolution

Given a view-graph G^l at l-th level with N_l nodes (*i.e.*, views) and node features are in rows of F_{in}^l, the local graph convolution layer is defined to update node features by considering relations among the neighboring nodes determined by kNN of camera coordinates. Given feature matrix $F_{in}^l \in R^{N_l \times d}$, a local graph convolution is defined as

$$F^l = \Psi(A^l F_{in}^l W^l; \theta_c^l) \tag{3}$$

where A^l represents the learnable $N_l \times N_l$ adjacency matrix of graph G^l, as defined in Eqn. (2), $W^l \in R^{d \times d}$ is the learnable weight matrix, and Ψ is non-linear transform consisting of a Batch Normalization [26] followed by a LeakyReLU [35] with parameters θ_c^l. Thus output F^l is still in $R^{N_l \times d}$. Using Eqn. (3), the input node features are firstly diffused by adjacency matrix A^l, then updated per-node by linear transform W^l, and followed by non-linear transform. Rows of F^l are the updated node features.

5.3. Non-local message passing

After local graph convolution, features in F^l are sent to non-local message passing operation to capture long-range relations among nodes in view-graph G^l. We define message from node v_i to v_j as a pairwise relation [38, 45]:

$$\boldsymbol{m}_{ij}^l = \Gamma([\boldsymbol{f}_i^l, \boldsymbol{f}_j^l]; \theta_m^l), i, j = 1, 2, ..., N_l \tag{4}$$

where $\boldsymbol{f}_i^l \in R^d$ is feature of i-th node, *i.e.*, the i-th row of F^l, $[\cdot, \cdot]$ denotes the concatenation of two vectors. Γ is a relation function with parameters θ_m^l aiming at exploring the relation between any paired views in the graph. We design it as a three-layer MLP with d hidden units and LeakyReLU in each layer, and it outputs message $\boldsymbol{m}_{ij} \in R^d$.

We further collect messages for node i from all the nodes in the graph, then update node feature by fusing the cumulated message $\boldsymbol{r}_i^l$ with original feature $\boldsymbol{f}_i^l$ as

$$\hat{\boldsymbol{f}}_i^l = \Omega([\boldsymbol{f}_i^l, \boldsymbol{r}_i^l]; \theta_f^l), \quad \text{where } \boldsymbol{r}_i^l = \sum_{j=1}^{N_l} \boldsymbol{m}_{ji}^l \tag{5}$$

Ω is a fusion function with parameters θ_f^l. It is designed as a one-layer MLP with Batch Normalization in implementation, and outputs the fused feature $\hat{\boldsymbol{f}}_i^l \in R^d$ for i-th node. Non-local message passing operation outputs feature matrix $\hat{F}^l$ with rows as node features in Eqn. (5).

By Eqn. (5), node features are updated considering pairwise relations over the whole graph, such that the updated features could incorporate messages from distant views, beyond local neighboring views in local graph convolution.

5.4. Selective view-sampling for graph coarsening

After updating node features of l-th level view-graph G^l, we then coarsen the graph to derive the view-graph G^{l+1} for $(l+1)$-th level. Graph coarsening is widely implemented by Farthest Point Sampling (FPS) in GCN [42], which samples a subset of views to build a coarsened graph to enlarge the receptive fields for GCN. We design a novel selective view-sampling strategy for graph coarsening.

Given input views with cameras coordinates $\{v_i^l\}_{i=1}^{N_l}$ over current graph G^l and sampling rate s, FPS iteratively samples a subset of views with camera coordinates $\{v_j\}_{j=1}^{N_{l+1}} \subset \{v_i^l\}_{i=1}^{N_l}$, $N_{l+1} = [N_l \times s]$ where $[\,]$ is a rounding function. FPS samples each new view with largest distance to already sampled set of views based on camera coordinates. Samples by FPS can keep the diversity of views, but can not guarantee that the sampled views are representative for downstream discriminative learning task.

To take advantage of FPS for sampling diverse views while overcoming its drawback, we propose a *selective view-sampling strategy* to select a set of representative views neighboring the views sampled by FPS using *view selector*. As illustrated in the bottom of Fig. 5, we first sample a subset of views $\{v_j\}_{j=1}^{N_{l+1}}$ by FPS as initialization. Given an initial view with camera coordinates v_j by FPS, we then select a sampled view by view-selector over kNN views of

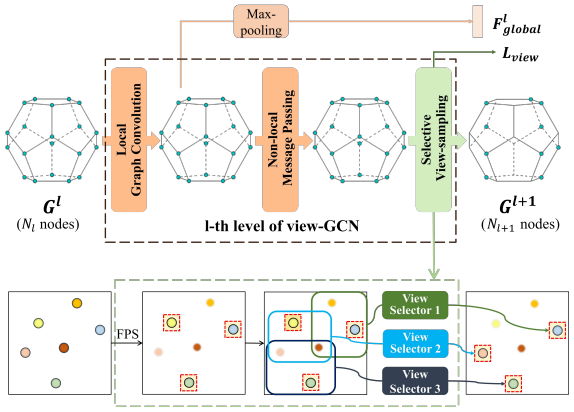


Figure 5. One level of view-GCN. It consists of local graph convolution, non-local message passing and selective view-sampling. We also present details of selective view-sampling at the bottom. Sampled views are shown as points in dashed rectangles.

this initial view, and the newly sampled view is with maximal response to view selector in local neighborhood. Then camera coordinates vector of sampled view is

$$v_j^{l+1} = \arg\max_{v_q \in \mathcal{N}(v_j)} \left(\max(V(\hat{\boldsymbol{f}}_{v_q}^l; \theta_v^{l,j})) \right) \quad (6)$$

for $j = 1, \cdots, N_{l+1}$. $V(\cdot) \in R^{N_C}$ is view selector with parameters $\theta_v^{l,j}$, outputting probabilities of a view belonging to N_C shape classes, and max operator indicates the maximal value in a vector. View selector is separately defined for different sampled views $j \in [1, N_{l+1}]$ and graph level indexed by l. For simplicity, view-selector V is defined as a two-layer MLP with $d/2$ hidden units, and its parameters are learned based on the training loss in Sect. 5.6.

By this strategy, we derive a coarsened graph G^{l+1} with graph nodes $\{v_j^{l+1}\}_{j=1}^{N_{l+1}}$ selected by view selectors. Each graph node is attached with its corresponding updated view feature after non-local message passing in Eqn. (5), and theses features can be represented as rows of feature matrix F_{in}^{l+1}, taken as input node features for next level $l+1$.

View-selectors are learnable components of view-GCN, each of which can be taken as a view template, learned to select discriminative view among local neighboring views.

5.5. Hierarchical network architecture

As shown is Fig. 5, one level of view-GCN is composed of consecutive local graph convolution, non-local message passing and selective view-sampling. For the l-level view-GCN, it embeds multi-view features on a graph G^l to output the updated features over a coarsened graph G^{l+1} with less number of views. We concatenate multiple levels of view-GCN to be a hierarchical deep architecture as shown in Fig. 2. To retain all the shape features in the hierarchy, in each level, we perform max-pooling on node features updated by local graph convolution to be a pooled descriptor

$$F_{global}^l = \text{maxpool}(\{\boldsymbol{f}_i^l\}_{i=1}^{N_l}), l = 0, 1, ..., L-1 \quad (7)$$

and the final global shape feature is the concatenation of all the pooled features at all levels: $\boldsymbol{F} = [F_{global}^0, \cdots, F_{global}^{L-1}]$, which is sent to training loss. We next present two versions of view-GCN with typical multi-view configurations. In all networks, $d = 512$, $s = 0.5$, and FPS always starts from the first graph node, *i.e.*, view, for different shapes.

View-GCN for 12-view circular configuration. As shown in Fig. 4 (a), the virtual cameras are regularly placed on a circular trajectory, and elevated with 30 degrees around the upright direction. The 12 views of a 3D shape construct a view-graph with 12 nodes and $k = 2$ for kNN. The view-graph is coarsened twice, then the view-GCN is a hierarchy over view-graphs with 12, 6 and 3 nodes.

View-GCN for 20-view dodecahedral configuration. We place virtual cameras on the vertices of a dodecahedron encompassing the object as shown in Fig. 4 (b). Taking 20 view features as a view-graph with 20 nodes and $k = 3$ in kNN. The view-GCN is defined over view-graph hierarchy with 20, 10, 5 nodes respectively.

5.6. Network training

Training loss. The overall training loss function consists of shape loss $\mathcal{L}_{shape}$ and view loss $\mathcal{L}_{view}$. Given global shape feature $\boldsymbol{F}$, it is sent to a classifier $\mathcal{C}$ with a FC layer having weights $W_c \in R^{Ld \times N_C}$, followed by a softmax layer. The total training loss is

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{shape}(\mathcal{C}(\boldsymbol{F}), y) \\ &+ \sum_{l=0}^{L-1} \sum_{j=1}^{N_{l+1}} \sum_{v_q \in N(v_j)} \mathcal{L}_{view}\left(V(\hat{\boldsymbol{f}}_{v_q}^l; \theta_v^{l,j}), y\right) \end{aligned} \quad (8)$$

where y is class label of shape, $\mathcal{L}_{shape}$ is cross-entropy loss based on global shape descriptor $\boldsymbol{F}$, $\mathcal{L}_{view}$ is cross-entropy loss defined for view-selectors enforcing that each view-selector can discriminate the shape category based on view features of a local neighborhood of views.

Trainable parameters. Trainable network parameters are denoted by Θ including $W^l, \theta_s^l, \theta_c^l, \theta_m^l, \theta_f^l, \theta_v^{l,i}$, for $l = 0, \cdots, L-1, i = 1, \cdots, N_l$ in different operations of view-GCN, and W_c in classifier $\mathcal{C}$. For the 20-view version of view-GCN, it has 73.4M parameters to learn, including 44.8M parameters of ResNet-18 (backbone view feature extraction network) and 28.6M parameters of our view-GCN. The backbone network parameters are also fune-tuned.

Training method. We train view-GCN in two steps similar to [48]. Firstly, the pre-trained view feature extractor, *e.g.*, ResNet-18 on ImageNet [11], is fine-tuned on all multi-view 2D images for classification, as discussed

in Sect. 5.1. Secondly, we train the whole architecture including backbone view feature extractor and view-GCN on training 3D shapes for shape recognition by end-to-end training. The gradients of loss w.r.t. parameters of view-GCN and view feature extraction network can be computed by auto-differentiation implemented by PyTorch [39].

Training details. When fine-tuning view feature extraction network, we use SGD optimizer with momentum, weight decay, batch size, epoch number and initial learning rate as $0.9, 10^{-2}, 400, 30, 10^{-2}$ respectively. The learning rate is reduced by half every 10 epochs. When training whole architecture, we also use SGD optimizer and change the learning rate to 10^{-3} and run in 15 epochs. Each batch contains 20 shapes with a total of 400 views for 20-views version of view-GCN, and 32 shapes with a total of 384 views for 12-views version of view-GCN. Following [22], we use a learning rate warm-up strategy, where the learning rate linearly increases from 0 to 10^{-3} in first epoch. Then the learning rate is reduced to 0 following a cosine quarter-cycle. Our code will be available on `https://github.com/weixmath/view-GCN`.

6. Experiments

We evaluate view-GCN for 3D shape classification and retrieval on synthetic and real datasets as follows.

ModelNet40 [51]. It consists of 12,311 3D shapes from 40 categories, with 9,843 training and 2,468 test objects for shape classification. There are different number of shapes across different categories.

ShapeNet Core55 [46]. It contains 51,162 3D models categorized into 55 classes, which are further divided into 203 sub-categories. The training, validation and test sets consist of 35764, 5133 and 10265 shapes respectively. Different classes have varying number of objects. We evaluate on the "normal" dataset, *i.e*., all of the shapes are consistently aligned and normalized to a unit length cube.

RGBD [31]. This is a real captured multi-view dataset containing both RGB and RGBD images of 300 household objects in 51 categories taken from multiple viewpoints. Each object is placed on a turntable and cameras elevate at approximately $30°$, $45°$ and $60°$ above the horizon.

6.1. Experiment for 3D shape classification

We first evaluate view-GCN on ModelNet40 for shape classification. The training takes 3 and 6 hours in first and second training stages on ModelNet40 using a NVIDIA GTX 1080 Ti GPU. We compare with diverse 3D object classification methods and mainly focus on view-based methods. Classification results are presented in Table 1. We achieve highest scores for both per class and per instance accuracies. Compared with traditional view-pooling methods like MVCNN [47], MVCNN-new [48], MHBN [55], GVCNN [15], and RCPCNN [50], our view-GCN achieves significantly higher accuracies by more than 3.4% per class and 2.6% per instance accuracies. 3D2SeqViews[19], SeqViews2SeqLabels [20], VERAM [8], and Ma et al. [34] all exploit relations on sequential views. Compared with them, view-GCN investigates relations of multi-view features over a hierarchy of view-graphs and improves the per class and per instance accuracies by more than 4.4% and 3.9% respectively. We also compare with methods based on points, voxels and mixed representations, including 3DShapeNets [51], VoxNet [37], VRN Ensemble [5], MVCNN-MultiRes [41], PointNet++ [42], Kd-Networks [30], RS-CNN [33], our view-GCN also outperforms them by more than 5.1% and 2.1% in two accuracies.

Among previous methods, RotationNet [28], which optimizes poses by rotation and investigates different view configurations, has achieved state-of-the-art performance.

Table 1. Shape classification accuracy (in %) on ModelNet40.

ModelNet40			
Method	Input	Per Class Acc.	Per Ins. Acc.
3DShapeNets [51]	Voxels	77.3	–
VoxNet [37]		83.0	–
VRN Ensemble [5]		–	95.5
MVCNN-MultiRes [41]		91.4	93.8
PointNet++ [42]	Points	–	91.9
Kd-Networks [30]		88.5	91.8
RS-CNN [33]		–	93.6
MVCNN [47]	Images	90.1	90.1
MVCNN-new [48]		92.4	95.0
MHBN [55]		93.1	94.7
GVCNN [15]		90.7	93.1
RCPCNN [50]		–	93.8
3D2SeqViews[19]		91.5	93.4
SeqViews2SeqLabels [20]		91.1	93.3
VERAM [8]		92.1	93.7
Ma *et al*. [34]		–	91.5
iMHL [56]		–	97.2
DeepCCFV [25]		–	92.5
HGNN [14]		–	96.7
EMV [13]		92.6	94.7
RN [53]		92.3	94.3
View-GCN (ResNet-18)	Images	**96.5**	**97.6**

Table 2. Comparison with RotationNet on ModelNet40 (in %).

Method	Backbone	Per Ins. Acc
RotationNet	AlexNet	96.4
View-GCN		**97.2**
RotationNet	ResNet-50	96.9
View-GCN		**97.3**

Table 3. Shape retrieval results (in %) on ShapeNet Core55 dataset.

ShapeNet Core55										
Method	microALL					macroALL				
	P@N	R@N	F1@N	mAP	NDCG	P@R	R@N	F1@N	mAP	NDCG
ZFDR	53.5	25.6	28.2	19.9	33.0	21.9	40.9	19.7	25.5	37.7
DeepVoxNet	79.3	21.1	25.3	19.2	27.7	59.8	28.3	25.8	23.2	33.7
DLAN	**81.8**	68.9	71.2	66.3	76.2	61.8	53.3	50.5	47.7	56.3
GIFT [2]	70.6	69.5	68.9	64.0	76.5	44.4	53.1	45.4	44.7	54.8
Improved GIFT [3]	78.6	77.3	76.7	72.2	82.7	59.2	65.4	58.1	57.5	65.7
ReVGG	76.5	80.3	77.2	74.9	82.8	51.8	60.1	51.9	49.6	55.9
MVFusionNet	74.3	67.7	69.2	62.2	73.2	52.3	49.4	48.4	41.8	50.2
CM-VGG5-6DB	41.8	71.7	47.9	54.0	65.4	12.2	**66.7**	16.6	33.9	40.4
MVCNN [47]	77.0	77.0	76.4	73.5	81.5	57.1	62.5	57.5	56.6	64.0
RotationNet [28]	81.0	80.1	79.8	77.2	**86.5**	60.2	63.9	59.0	58.3	65.6
View-GCN (ResNet-18)	**81.8**	**80.9**	**80.6**	**78.4**	85.2	**62.9**	65.2	**61.1**	**60.2**	**66.5**

For fair comparison, as shown in Table 2, with the same AlexNet as backbone network for view feature extraction and 20 views configuration, our view-GCN achieves 0.8% per instance accuracy higher than RotationNet.

6.2. Experiment for 3D shape retrieval

ShapeNet Core55 [46] is a challenging 3D dataset for shape retrieval. We train our view-GCN on 20 input views for shape classification same as ModelNet40 for shape classification. For shape retrieval, given each query object, all the objects with the same predicted class label are firstly taken as retrieved shapes, and the retrieval rank is based on ranking the probability scores for class label prediction. As required by the challenge [46], top 1000 retrieved objects for each category are taken as retrieval results.

We compare view-GCN with diverse methods that attended the track of SHREC'17 for Large-Scale 3D Shape Retrieval [46] on ShapeNet Core55, including multi-view based methods such as GIFT [2], Improved-GIFT, ReVGG, MVFusionNet, CM-VGG55-6DB, MVCNN[47] and RotationNet [28], and voxel-based methods such as ZFDR, DeepVoxelNet and DLAN. For more details on these methods and accuracy metrics, please refer to [46].

As shown in Table 3, our view-GCN achieves highest accuracies for micro-averaged P@N, R@N, F1@N, mAP and macro-averaged P@N, F1@N, mAP and NDCG. View-GCN outperforms current state-of-the-art method RotationNet on this dataset on all metrics except micro-averaged NDCG. Compared with other methods such as GIFT [2], Improved-GIFT and MVCNN [47], view-GCN also achieves significantly higher accuracies, *e.g.*, it achieves 4.9% (in mAP for microALL) higher than MVCNN, a baseline method that conducts max-pooling of multi-view features, while our view-GCN aggregates multi-view features hierarchically on view-graphs.

6.3. Experiment on real multi-view image dataset

We also evaluate our view-GCN for shape classification on RGBD dataset [31], which is a dataset with real captured multi-view images. We use the same experimental setup as in [28], *i.e.*, for each object, we uniformly take 12 RGB multi-view images captured by cameras on a circle with 45° camera elevation angle. We perform ten-fold cross-validation to report average results as suggested in [31].

Table 4. Comparison of classification accuracies (in %) on RGBD.

RGBD		
Method	#View	Per Ins. Acc
MDSICNN [1]	≥ 120	89.6
CFK [9]	≥ 120	86.8
MMDCNN [43]	≥ 120	89.5
MVCNN (AlexNet) [47]	12	86.1
RotationNet (AlexNet) [28]	12	89.3
View-GCN (AlexNet)	12	91.9
View-GCN (ResNet-18)	12	**94.3**
View-GCN (ResNet-50)	12	93.9

As shown in Table 4, our view-GCNs achieve best results in classification accuracy. By using same backbone network of AlexNet, our view-GCN (AlexNet) outperforms MVCNN (AlexNet) [47] by 5.8% in per instance accuracy. View-GCN (AlexNet) also outperforms RotationNet (AlexNet) [28] by 2.6% in accuracy. Our view-GCNs significantly exceed performances of MDSICNN [1], MMD-CNN [43] that take more RGB images as inputs. Using more powerful backbone ResNet-18, view-GCN (ResNet-18) achieves highest accuracy of 94.3%. View-GCN with backbone network of ResNet-18 works marginally better than that with ResNet-50. These results show that view-GCN also works well for real multi-view images.

Table 5. Results (in %) of variants of view-GCN for shape classification on ShapeNet Core55 with different architectures.

Method	Per Class Acc.	Per Ins. Acc.
Baseline	76.7	88.9
View-GCN (w/o LGC)	78.8	90.6
View-GCN (w/o NLMP)	77.7	90.5
View-GCN-FPS	78.2	90.3
View-GCN-A1	79.2	90.7
View-GCN-A2	79.1	90.5
View-GCN-L1	78.5	89.9
View-GCN-L2	78.6	90.6
View-GCN (w/o view loss)	79.7	90.7
View-GCN (NLMP)	78.3	90.4
View-GCN	**79.8**	**90.9**

6.4. Experimental analysis on view-GCN

We next justify effect of each component of view-GCN on ShapeNet Core55 [46] for classification.

Results of various architectures of view-GCN are in Table 5. All nets take ResNet-18 as backbone network. "Baseline" is MVCNN-new [48] method that uses max-pooled feature for global shape descriptor. Compared with it, our view-GCN achieves 79.8% in per class accuracy and 90.9% in per instance accuracy, compared to 76.7% and 88.9% by "Baseline". By removing local graph convolution from view-GCN, view-GCN (w/o LGC) achieves 78.8% and 90.6% which are 1.0% and 0.3% lower than full version in two accuracies. View-GCN without non-local message passing, *i.e.*, view-GCN (w/o NLMP), achieves 2.1% and 0.4% lower per class and per instance accuracies. They demonstrate positive contributions of our network modules.

Selective view-sampling *vs*. FPS. To evaluate the effect of selective view-sampling, we replace the selective view-sampling by simple FPS (View-GCN-FPS) for graph coarsening. As shown in Table 5, compared with view-GCN using FPS, view-GCN using selective view-sampling achieves improvements of 1.6% and 0.6% respectively for per class and per instance accuracies. We also perform same experiment on unaligned ModelNet40 [51], and our view-GCN with selective view-sampling achieves 0.9% and 0.8% higher in two accuracies than view-GCN using FPS, which demonstrates effectiveness of selective view-sampling strategy for selecting representative views for graph coarsening.

Effect of view loss. We further evaluate effect of view loss $\mathcal{L}_{view}$ which is used to enforce discriminative ability of view-selectors as mentioned in Sect. 5.6. By training same network without view loss $\mathcal{L}_{view}$, view-GCN (w/o view-loss) achieves marginally lower scores (0.1% and 0.2% lower in two accuracies), showing that even without explicitly imposing view loss, view-GCN can learn the parameters of view-selectors and still achieves 1.5% and 0.4% higher in two accuracies than graph coarsening using FPS.

Effect of learning affinity matrix. For each level view-graph, we learn adjacency matrix by Eqn. (2). To justify its necessity, we compare baselines of view-GCN-A1 and view-GCN-A2 whose element of affinity matrix is respectively defined as $A_{ij} = \mathbb{I}\{v_j \in \mathcal{N}(v_i)\}$, and $A_{ij} = e^{-\|v_i - v_j\|_2} \cdot \mathbb{I}\{v_j \in \mathcal{N}(v_i)\}$. View-GCN achieves improvements of (0.6%, 0.2%) and (0.7%, 0.4%) respectively compared with two baselines in two accuracies in brackets.

Effect of hierarchical structure. Our view-GCN for 20 input views is defined over a hierarchy of view-graphs with 20, 10, 5 nodes. We also compare with view-GCN-L1 and view-GCN-L2 defined over a 1-level view-graph (20 nodes) and 2-levels view-graphs (20 and 10 nodes). As shown in Table 5, view-GCN-L1 achieves 1.8% and 1.0% higher accuracy than "Baseline", and view-GCN-L2 further improves results by 0.1% and 0.7%. The final 3-levels view-GCN achieves 1.2% and 0.3% improvements than view-GCN-L2, showing effectiveness of hierarchical structure.

Effect of sampling rate. By increasing sampling rate s to 0.6 and 0.7, the view-GCNs are deeper with 4 and 5 levels having [20,12,7,4], [20,14,10,7,5] nodes respectively, and accuracies slightly drop by (0.1%,0.2%) and (0.3%,0.3%) in per class and per instance accuracies.

Selection of layer for constructing global feature. View-GCN performs max-pooling on node features after local graph convolution in each level to construct global shape descriptor. We also present result of view-GCN (NLMP) having the same architecture as view-GCN except that max-pooling is performed on the node features updated by non-local message passing in each level. View-GCN (NLMP) achieves 1.5% and 0.5% lower accuracies than view-GCN.

Extension to irregular view configuration. View-GCN can also be flexibly extended to irregular view configurations. Taking Fig. 4 (c) as an example, this configuration is based on randomly selected 12 views from Fig. 4 (b), and each view is randomly perturbed in coordinates. We design our view-GCN with three levels of 12, 6, 3 nodes, and it achieves 85.3% per class and 89.5% per instance accuracies, 4.2% and 1.9% higher than MVCNN-new [48].

7. Conclusion

We proposed a novel graph convolutional network for 3D shape recognition. We model multiple views of a shape by view-graph, and develop a novel GCN on hierarchical view-graphs to learn global shape descriptor. Extensive experiments justify its effectiveness. In future work, we plan to apply view-GCN to multi-modality feature fusion.

Acknowledgment This work was supported by NSFC (11971373, 11690011, U1811461, 61721002) and National Key R&D Program 2018AAA0102201.

References

[1] Umar Asif, Mohammed Bennamoun, and Ferdous A Sohel. A multi-modal, discriminative and spatially invariant cnn for rgb-d object labeling. *IEEE PAMI*, 40(9):2051–2065, 2017.

[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*, 2016.

[3] Song Bai, Xiang Bai, Zhichao Zhou, Zhaoxiang Zhang, Qi Tian, and Longin Jan Latecki. Gift: Towards scalable 3d shape retrieval. *IEEE TMM*, 19(6):1257–1271, 2017.

[4] Davide Boscaini, Jonathan Masci, Emanuele Rodolà, and Michael Bronstein. Learning shape correspondence with anisotropic convolutional neural networks. In *NeurIPS*, 2016.

[5] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*, 2016.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

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

[8] Songle Chen, Lintao Zheng, Zhang Yan, Zhixin Sun, and Xu Kai. Veram: View-enhanced recurrent attention model for 3d shape classification. *IEEE TVCG*, PP(99):1–1, 2018.

[9] Yanhua Cheng, Rui Cai, Xin Zhao, and Kaiqi Huang. Convolutional fisher kernels for rgb-d object recognition. In *3DV*, pages 135–143. IEEE, 2015.

[10] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, pages 3844–3852, 2016.

[11] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Li. Imagenet: a large-scale hierarchical image database. In *CVPR*, 2009.

[12] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*, pages 2224–2232, 2015.

[13] Carlos Esteves, Yinshuang Xu, Christine Allen-Blanchette, and Kostas Daniilidis. Equivariant multi-view networks. In *ICCV*, 2019.

[14] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. Hypergraph neural networks. In *AAAI*, volume 33, pages 3558–3565, 2019.

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

[16] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *ICML*, pages 1263–1272, 2017.

[17] Nate Hagbi, Oriel Bergig, Jihad El-Sana, and Mark Billinghurst. Shape recognition and pose estimation for mobile augmented reality. *IEEE TVCG*, 17(10):1369–1379, 2011.

[18] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, pages 1024–1034, 2017.

[19] Zhizhong Han, Honglei Lu, Zhenbao Liu, Chi-Man Vong, Yu-Shen Liua, Matthias Zwicker, Junwei Han, and CL Philip Chen. 3d2seqviews: Aggregating sequential views for 3d global feature learning by cnn with hierarchical attention aggregation. *IEEE TIP*, 2019.

[20] Zhizhong Han, Mingyang Shang, Zhenbao Liu, et al. Seqviews2seqlabels: Learning 3d global features via aggregating sequential views by rnn with attention. *IEEE TIP*, 28(2):658–672, 2019.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[22] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *CVPR*, pages 558–567, 2019.

[23] Xinwei He, Tengteng Huang, Song Bai, and Xiang Bai. View n-gram network for 3d object retrieval. In *ICCV*, 2019.

[24] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[25] Zhengyue Huang, Zhehui Zhao, Hengguang Zhou, Xibin Zhao, and Yue Gao. Deepccfv: Camera constraint-free multi-view convolutional neural network for 3d object retrieval. In *AAAI*, volume 33, pages 8505–8512, 2019.

[26] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[27] Jianwen Jiang, Di Bao, Ziqiang Chen, Xibin Zhao, and Yue Gao. Mlvcnn: Multi-loop-view convolutional neural network for 3d shape retrieval. In *AAAI*, volume 33, pages 8513–8520, 2019.

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

[29] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

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

[31] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, pages 1817–1824. IEEE, 2011.

[32] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *NeurIPS*, 2018.

[33] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019.

[34] Chao Ma, Yulan Guo, Jungang Yang, and Wei An. Learning multi-view representation with lstm for 3-d shape recognition and retrieval. *IEEE TMM*, 21(5):1169–1182, 2018.

[35] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML*, volume 30, page 3, 2013.

[36] Jonathan Masci, Davide Boscaini, Michael Bronstein, and Pierre Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *ICCV-Workshops*, pages 37–45, 2015.

[37] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *IROS*, 2015.

[38] Rasmus Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks. In *NeurIPS*, 2018.

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary Devito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NeurIPS-Workshops*, 2017.

[40] T Pylvanainen, Kimmo Roimela, Ramakrishna Vedantham, J Itaranta, and R Grzeszczuk. Automatic alignment and multi-view segmentation of street view data using 3d shape priors. *3DPVT*, 737:738–739, 2010.

[41] 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*, 2016.

[42] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017.

[43] Mohammad Muntasir Rahman, Yanhao Tan, Jian Xue, and Ke Lu. Rgb-d object recognition with multimodal deep convolutional neural networks. In *ICME*, pages 991–996. IEEE, 2017.

[44] Heather Richards-Rissetto, Fabio Remondino, Giorgio Agugiaro, Jennifer von Schwerin, Jim Robertsson, and Gabrio Girardi. Kinect and 3d gis in archaeology. In *International Conference on Virtual Systems and Multimedia*, 2012.

[45] 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 *NeurIPS*, 2017.

[46] Manolis Savva, Fisher Yu, Hao Su, Asako Kanezaki, Takahiko Furuya, Ryutarou Ohbuchi, Zhichao Zhou, Rui Yu, Song Bai, Xiang Bai, et al. Large-scale 3d shape retrieval from shapenet core55: Shrec'17 track. In *Proceedings of the Workshop on 3D Object Retrieval*, 2017.

[47] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik G Learnedmiller. Multi-view convolutional neural networks for 3d shape recognition. In *ICCV*, 2015.

[48] Jong-Chyi Su, Matheus Gadelha, Rui Wang, and Subhransu Maji. A deeper look at 3d shape classifiers. In *Second Workshop on 3D Reconstruction Meets Semantics, ECCV*, 2018.

[49] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

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

[51] 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*, 2015.

[52] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. Spidercnn: Deep learning on point sets with parameterized convolutional filters. In *ECCV*, 2018.

[53] Ze Yang and Liwei Wang. Learning relationships for multi-view 3d object recognition. In *ICCV*, 2019.

[54] Mohsen Yavartanoo, Eu Young Kim, and Kyoung Mu Lee. Spnet: Deep 3d object classification and retrieval using stereographic projection. In *ACCV*, 2018.

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

[56] Zizhao Zhang, Haojie Lin, Xibin Zhao, Rongrong Ji, and Yue Gao. Inductive multi-hypergraph learning and its application on view-based 3d object classification. *IEEE TIP*, 27(12):5957–5968, 2018.