Joint Multi-view 2D Convolutional Neural Networks for 3D Object Classification

Jinglin Xu[1] , **Xiangsen Zhang**[1] , **Wenbin Li**[2] , **Xinwang Liu**[3] and **Junwei Han**[1*]
[1]Northwestern Polytechnical University, Xi'an, China
[2]Nanjing University, Nanjing, China
[3]National University of Defense Technology, Changsha, China
{xujinglinlove, liwenbin.nju, junweihan2010}@gmail.com, xszhang@mail.nwpu.edu.cn,
xinwangliu@nudt.edu.cn

Abstract

Three-dimensional (3D) object classification is widely involved in various computer vision applications, e.g., autonomous driving, simultaneous localization and mapping, which has attracted lots of attention in the committee. However, solving 3D object classification by directly employing the 3D convolutional neural networks (CNNs) generally suffers from high computational cost. Besides, existing view-based methods cannot better explore the content relationships between views. To this end, this work proposes a novel multi-view framework by jointly using multiple 2D-CNNs to capture discriminative information with relationships as well as a new multi-view loss fusion strategy, in an end-to-end manner. Specifically, we utilize multiple 2D views of a 3D object as input and integrate the intra-view and inter-view information of each view through the view-specific 2D-CNN and a series of modules (outer product, view pair pooling, 1D convolution, and fully connected transformation). Furthermore, we design a novel view ensemble mechanism that selects several discriminative and informative views to jointly infer the category of a 3D object. Extensive experiments demonstrate that the proposed method is able to outperform current state-of-the-art methods on 3D object classification. More importantly, this work provides a new way to improve 3D object classification from the perspective of fully utilizing well-established 2D-CNNs.

1 Introduction

Object classification is a critical task in computer vision applications. It is the task of classifying the objects in the received images and can be helpful in future tasks such as object detection and tracking. Traditional ways of object classification extract features (such as HOG, SIFT, SURF) or descriptors first and then use classifiers (e.g., SVM, Bayes model, graph propagation) to classify objects.

The accuracy of object classification can be improved by making good use of multiple different views of a target object [Paletta and Pinz, 2000]. Recent significant advances in image recognition and 3D object model collection make it possible to learn the multi-view representations of 3D objects. This has greatly facilitated the rapid development of 3D object classification, making it an important field in 3D computer vision, with a variety of applications, e.g., autonomous driving, intelligent robots, and virtual reality. Inspired by the success of deep learning in 2D visions, numerous deep learning-based 3D object classification methods [Wu *et al.*, 2015; Su *et al.*, 2015; Maturana and Scherer, 2015; Qi *et al.*, 2016; Brock *et al.*, 2016; Qi *et al.*, 2017; Yavartanoo *et al.*, 2018] are proposed recently, achieving significantly better performance compared to traditional handcrafted feature-based methods [Guo *et al.*, 2013; Guo *et al.*, 2016] and single view object classification methods [Krizhevsky *et al.*, 2012; Simonyan and Zisserman, 2015].

Considering three kinds of inputs are generally used to do the 3D object classification: multiple-view images, 3D voxel grids, and point clouds, we briefly review the previous works from the view-based methods [Su *et al.*, 2015; Johns *et al.*, 2016; Kanezaki *et al.*, 2018; Feng *et al.*, 2018; Wang *et al.*, 2019], the volume-based methods [Wu *et al.*, 2014; Maturana and Scherer, 2015; Brock *et al.*, 2016; Ren *et al.*, 2017] and the pointset-based methods [Qi *et al.*, 2017; Klokov and Lempitsky, 2017; Li *et al.*, 2018], respectively.

For the view-based methods, they project 3D objects into multiple 2D views and then utilize the features extracted from the 2D-CNN for classification. For example, MVCNN [Su *et al.*, 2015] utilizes multiple 2D views rendered by 3D objects as inputs for 2D-CNNs. A pairwise decomposition method is proposed in [Johns *et al.*, 2016], which outperforms MVCNN at the expense of increased training costs. This pairwise decomposition method uses two CNNs for selecting view pair and predicting pairwise label, respectively, each of which uses a CNN-M [Chatfield *et al.*, 2014] and has to be trained separately. As an extension of MVCNN, RotationNet [Kanezaki *et al.*, 2018] explores multiple views from different angles, takes a part of the entire multi-view image of an object as input, and infers the category of the object through the rotation. GVCNN [Feng *et al.*, 2018] considers the group information of multiple views and proposes the group-view CNNs, which groups the view-level features

*Contact Author

together to generate a group-level feature, and then merges the group-level features to obtain object-level features. The recurrent clustering and pooling layer introduced in [Wang *et al.*, 2019] is designed to combine the multi-view features, which provides more discriminative capabilities for 3D object classification. This work only pools information across similar views in contrast to MVCNN.

The volume-based approaches directly apply a 3D-CNN on voxelized shapes. Specifically, 3DShapeNets [Wu *et al.*, 2014] proposes to use convolutional Deep Belief Network (DBN) to describe the 3D geometry as a probability distribution on a 3D voxel grid. VoxNet [Maturana and Scherer, 2015] extends the 2D convolutional kernels to the 3D convolutional kernels. VRN Ensemble [Brock *et al.*, 2016] presents deep ConvNet architectures for modeling generative and discriminative voxel and explores the challenging problems of voxel-based representations. 3D-A-Nets [Ren *et al.*, 2017] develops a 3D adversarial network to solve the challenging problems of processing 3D volume data efficiently. Nevertheless, these 3D convolution-based methods have high computational complexity and GPU memory consumption.

The pointset-based methods directly take the unordered point sets as input. PointNet [Qi *et al.*, 2017] provides a unified framework to learn the global and local point features for the 3D classification tasks on the raw point clouds without any voxelization or rendering. Kd-Net [Klokov and Lempitsky, 2017] proposes a deep learning architecture that is capable of producing representations on point clouds working for different 3D data recognition tasks. SO-Net [Li *et al.*, 2018] explicitly constructs the spatial distribution of the input points and systematically adjusts the overlap of the receiving fields to extract hierarchical features on the point clouds.

Among the above three kinds of methods, the view-based methods usually perform better than the other two kinds of methods [Qi *et al.*, 2016; Feng *et al.*, 2018]. For one thing, the view-based methods can easily obtain more views from the 3D CAD model compared to other methods. For the other thing, the well-established 2D models (e.g., VGG, GoogLeNet, and ResNet) can be exploited for the powerful view representation. Therefore, in this paper, our work would classify the 3D objects in a view-based manner with a 2D-CNN architecture to achieve state-of-the-art results for multi-view object classification.

One straightforward solution to multi-view 3D object classification using 2D-CNN would be to simply concatenate all the views (represented as features) of an object as a single-view input. However, this concatenated input may reduce the interpretability of intra-view information of different views. Although some existing methods perform 2D-CNNs on views separately and aggregate them in a pooling layer, such pooling methods usually ignore the content relationships among views. To address these issues, we provide a novel idea for learning the discriminative *intra-view* information simultaneously, capturing the content relationship among views (*inter-view* information), and integrating these two kinds of information using a new *multi-view loss fusion strategy* for classifying 3D objects in an end-to-end way.

In this paper, we propose a novel multi-view framework for classifying 3D objects. Specifically, for the *intra-view* information, we utilize multiple 2D images (views) of a 3D object as input and extract the high-level intra-view information using multiple 2D-CNNs separately. This is helpful to explore the intrinsic attributed information for each view. For the *inter-view* information, we utilize the intra-view information of one view and that of all other different views to calculate the outer products, which obtains the correlation matrices between different attributes of each view pair. Then the enhanced correlation matrix is captured by the maximized operation at the corresponding locations of obtained correlation matrices in the direction of different view pairs. Furthermore, we apply 1D convolution and fully connected (FC) transformation over the enhanced correlation matrix to gain high-level inter-view information of each view, which is helpful to describe the content relationship across views. After obtaining the above information, we concatenate and feed them into the view-specific FC layer, which obtains the view-specific loss and label prediction. For the *multi-view loss fusion strategy*, we formulate a ℓ_0 constrained optimization problem with regard to the weights of multiple views and obtain the optimal weight distribution. This is beneficial to select some discriminative and informative views through the high weights and utilize their corresponding predictions to make a joint decision.

The main contributions of this work are summarized as follows:

- We propose a novel multi-view framework that captures the discriminative information with relationships across views for different views and designs a view ensemble mechanism via a multi-view loss fusion strategy, for classifying 3D objects in an end-to-end manner.
- We develop the discriminative information with relationships by integrating the intra-view and inter-view information, where the latter is generated by applying 1D convolution and FC transformation over the enhanced correlation matrix which is obtained by the outer product and view pair pooling. In addition, we design a novel multi-view loss fusion strategy by solving a ℓ_0 constraint optimization to make a joint decision for inferring the category.
- Extensive experimental results show that our proposed method can achieve better classification accuracy than most of the existing state-of-the-art methods on the ModelNet40 dataset.

2 Related Work

In recent years, deep CNNs have captured the most significant advance, especially for image classification, which classifies millions of images into thousands of categories. In contrast to the above single-view deep CNNs, multi-view CNNs considers learning convolutional representations in the setting where multiple views of data are available. It attempts to integrate discriminative information from different views, which generates more comprehensive representations for subsequent learning.

For example, in MVCNN [Su *et al.*, 2015], multi-view images obtained by the 3D rotations are passed through a shared

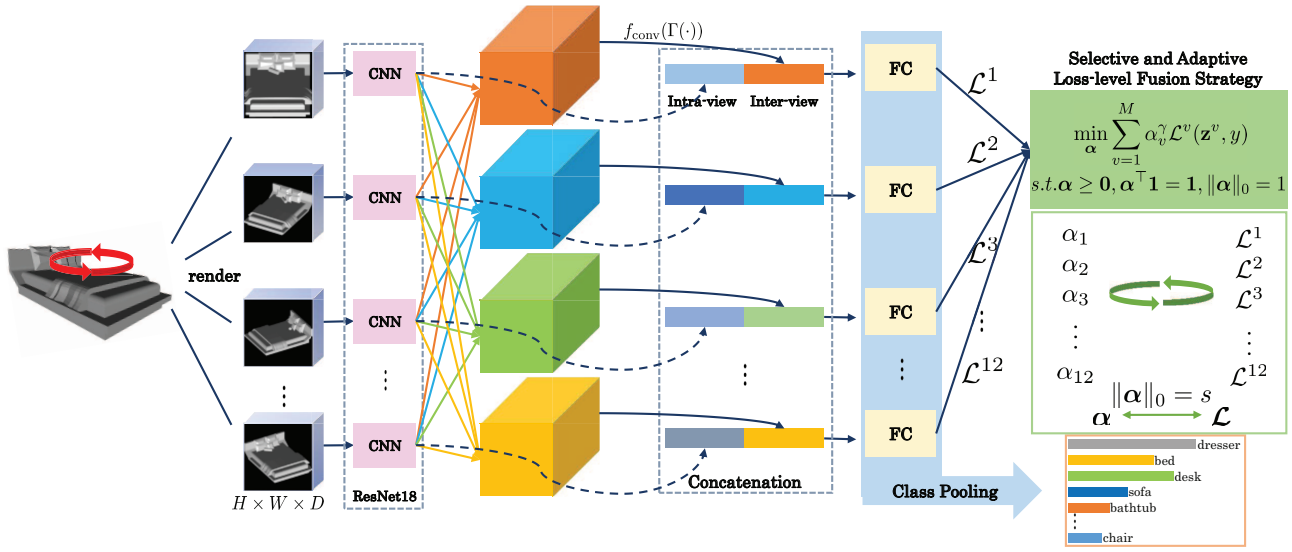

Figure 1: The architecture of our method. All branches in the first part of the network (i.e., CNN and $f_{\text{conv}}(\Gamma(\cdot))$) capture the intra-view and inter-view information of each view. The remaining part of the network (i.e., FC) is used to obtain the corresponding loss and label prediction of each view separately. The constrained optimization problem with regard to $\boldsymbol{\alpha}$ is formulated to obtain the optimal weight distribution and select discriminative predictions of views to make a joint decision.

CNN separately, fusing at a view-pooling layer and feeding into another CNN. However, the drawback of MVCNN is that its pooling layer ignores the divergence between different views, where some of the views are distinctive whereas others have common information.

GVCNN [Feng *et al.*, 2018] introduces the view, the group, and the shape level descriptor and provides a grouping scheme to divide the views in terms of the discrimination scores. However, the setting of the thresholds of grouping weight in the grouping module is unable to be guided by more discriminative information.

3 The Proposed Method

In this section, we illustrate the proposed method in detail, which is a joint multi-view 2D-CNNs learning framework to integrate the intra-view and inter-view information of the 3D objects by the multi-view convolutional representation with multi-loss fusion.

3.1 Formulation

Multiple Intra-view Information Extraction

The input data of the proposed method is rendered by multiple 2D views of a 3D object, which belongs to the view-based method. According to the previous works [Su *et al.*, 2015; Johns *et al.*, 2016; Feng *et al.*, 2018], 12 rendered views are created by placing 12 virtual cameras around the mesh every 30 degrees. The reason for rendering from such more viewpoints is that we do not exactly know which one can yield good representative views of an object. We make use of multiple 2D views to describe a 3D object, one 2D image per view. It is found that the multi-view representation contains rich information of 3D objects and can be applied to various practical tasks.

For the CNN features, we use the ResNet-18 [He *et al.*, 2016] as the base architecture which consists of 17 convolutional layers followed by one fully connected (FC) layer, to capture the intra-view information for each view. The ResNet-18 is pre-trained on ImageNet images from 1000 categories and then is fine-tuned on all 2D views of 3D objects in the training set. The CNN features can capture the high-level information for each view, which yields better performance on classification compared with some previous descriptors [Kazhdan *et al.*, 2003].

Multiple Inter-view Information Calculation

Based on the above subsection, given a 3D object, we first take a set of 2D input images captured from different angles, and each image is passed through a 2D-CNN to get the high-level representation in the view level.

Supposed that $\boldsymbol{x}^v \in \mathbb{R}^{H \times W \times D}$ and $\boldsymbol{x}^v_{\text{intra}} = f_{\text{cnn}}(\boldsymbol{x}^v) \in \mathbb{R}^{D_{\text{intra}}}$ are the input image and the learned features before FC layer by CNN from the v-th view, respectively, where H, W, and D denote the height, width, and channel. For the v-th view, we define a set S_v which contains different view pairs with respect to the v-th view, that is,

$$S_v|_{v=1}^{M} = \{(v, \bar{v})\}_{\bar{v}=\{1,\cdots,M\}\setminus v}|_{v=1}^{M}, \tag{1}$$

where M is the number of 2D input images and $(v, \bar{v}) = (\bar{v}, v)$. Therefore, the proposed 'inter-view' information for the v-th view $\boldsymbol{x}^v_{\text{inter}}$ across views can be calculated by using the outer product, view-pair pooling, and 1D convolution. That is:

$$\boldsymbol{x}^{v,\bar{v}}_{\text{en}} = \boldsymbol{x}^v_{\text{intra}} \otimes \boldsymbol{x}^{\bar{v}}_{\text{intra}} \tag{2}$$

$$\boldsymbol{x}^{S_v}_{\text{en}} = \{\boldsymbol{x}^{v,\bar{v}}_{\text{en}}\}_{\bar{v}=\{1,\cdots,M\}\setminus v} \tag{3}$$

$$\boldsymbol{x}^v_{\text{inter}} = f_{\text{conv}}(\Gamma(\boldsymbol{x}^{S_v}_{\text{en}})) \tag{4}$$

where $\boldsymbol{x}_{\text{en}}^{v,\bar{v}} \in \mathbb{R}^{D_{\text{intra}} \times D_{\text{intra}}}$ denotes the outer product of a view pair $(v, \bar{v})$, which captures correlations by multiplying each element of $\boldsymbol{x}_{\text{intra}}^{v}$ by each element of $\boldsymbol{x}_{\text{intra}}^{\bar{v}}$. Extending to all the view pairs of the v-th view, $\boldsymbol{x}_{\text{en}}^{S_v} \in \mathbb{R}^{D_{\text{intra}} \times D_{\text{intra}} \times (M-1)}$ collects the correlation information of the v-th view with respect to other $M-1$ views. Furthermore, $\Gamma(\boldsymbol{x}_{\text{en}}^{S_v}) \in \mathbb{R}^{D_{\text{intra}} \times D_{\text{intra}}}$ maximizes the correlations of $M-1$ view pairs in S_v along the direction of different view pairs for the v-th view, where Γ is the view pair pooling operation. Finally, the high-level inter-view information $\boldsymbol{x}_{\text{inter}}^{v} \in \mathbb{R}^{D_{\text{inter}}}$ is generated by applying f_{conv} over $\Gamma(\boldsymbol{x}_{\text{en}}^{S_v})$, which consists of two steps that transforming each row of $\Gamma(\boldsymbol{x}_{\text{en}}^{S_v})$ into a K-dimension vector by applying a 1D convolution (kernel_size=1) and concatenating D_{intra} K-dimension vectors to project into a D_{inter}-dimension vector (i.e., $\boldsymbol{x}_{\text{inter}}^{v}$) through an FC layer.

Multi-view Loss Fusion Strategy with ℓ_0 Constraint

After that, we combine $\boldsymbol{x}_{\text{intra}}^{v}$ and $\boldsymbol{x}_{\text{inter}}^{v}$ by a concatenated operation and then feed it into an FC layer to obtain the corresponding loss and label prediction of each view. That is,

$$\boldsymbol{x}_{\text{con}}^{v} = f_{\text{cat}}(\boldsymbol{x}_{\text{intra}}^{v}, \boldsymbol{x}_{\text{inter}}^{v}) \tag{5}$$

$$\boldsymbol{z}^{v} = f_{\text{fc}}(\boldsymbol{x}_{\text{con}}^{v}) \tag{6}$$

where $\boldsymbol{x}_{\text{con}}^{v} \in \mathbb{R}^{(D_{\text{intra}}+D_{\text{inter}})}$ denotes the comprehensive information of each view and $\boldsymbol{z}^{v} \in \mathbb{R}^{C}$ is produced by f_{fc} with input $\boldsymbol{x}_{\text{con}}^{v}$, indicating the probability distribution over the possible classes for each view, and C is the number of categories.

Next, we propose a novel adaptive-weighting loss fusion strategy with proper sparseness for multiple predictions $\boldsymbol{z}^{v}|_{v=1}^{M}$ to make a joint decision and implement the multi-view 3D object classification, which can be described as,

$$\min_{\boldsymbol{\alpha}^{\top}\mathbf{1}=1, \boldsymbol{\alpha} \geq \mathbf{0}, \|\boldsymbol{\alpha}\|_0 = s} \sum_{v=1}^{M} \alpha_v^{\gamma} \mathcal{L}^{v}(\boldsymbol{z}^{v}, y) \tag{7}$$

where

$$\mathcal{L}^{v}(\boldsymbol{z}^{v}, y) = -\log\left(\frac{\exp(z_y^{v})}{\sum_{o=1}^{C}\exp(z_o^{v})}\right), \tag{8}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{M}$ is a weight vector corresponding to multiple views, $y \in \mathbb{R}$ denotes the common label information of all the views for an object, and $\mathcal{L}^{v}(\boldsymbol{z}^{v}, y) \in \mathbb{R}$ is the cross-entropy loss of the v-th view. $\gamma > 1$ is the power exponent parameter of the weight α_v, which adjusts the weight distribution of different views flexibly and avoids the trivial solution of $\boldsymbol{\alpha}$ during the classification. $\|\boldsymbol{\alpha}\|_0 = s$ is used to constrain the sparseness of the weight vector $\boldsymbol{\alpha}$, where $s \in \mathbb{N}_+$ denotes the number of nonzero elements in $\boldsymbol{\alpha}$. Crucially, the ℓ_0-norm constraint is able to capture the global relations among different views and is able to achieve view-wise sparseness such that only a few discriminative and informative views are selected during the optimization to make decisions.

In summary, we design a novel multi-view framework based on multiple 2D-CNNs, shown in Figure 1. Each view of the 3D object is passed through CNN separately to obtain the intra-view information. Then, the intra-view information of different views can generate the corresponding high-level inter-view information by a series of operations, i.e., outer product, view pair pooling, and 1D convolution as well as FC transformation. All branches of the network, i.e., CNNs, share the same parameters. After that, the intra-view and inter-view information of each view are concatenated and fed into the FC layer, to obtain the corresponding loss and prediction. In addition, we formulate a constrained optimization problem with regard to the weights for multiple views, which can obtain the optimal weight distribution. Furthermore, the optimal weight distribution learned in the training stage is used to guide the testing stage, which can make a joint decision in the class pooling by the views with high weights.

3.2 Optimizing Weights of Multiple Views

Learning the weight α_v of each view assigns the discriminative and informative view with a higher weight. Therefore, we optimize $\boldsymbol{\alpha}$ by solving problem (7).

We define a function $\mathcal{P}(\cdot)$ on the loss vector $\boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)$,

$$\mathcal{P}(\boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)) = \boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)\boldsymbol{P} \tag{9}$$

where $\boldsymbol{\mathcal{L}}(\boldsymbol{z}, y) = [\mathcal{L}^{1}(\boldsymbol{z}^{1}, y), \cdots, \mathcal{L}^{M}(\boldsymbol{z}^{M}, y)]$ and $\boldsymbol{P}$ is a permutation matrix which results in the elements of $\boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)$ along the ascending order, i.e., $\mathcal{L}^{\mathcal{P}(1)}(\boldsymbol{z}^{\mathcal{P}(1)}, y) \leq \cdots \leq \mathcal{L}^{\mathcal{P}(M)}(\boldsymbol{z}^{\mathcal{P}(M)}, y)$. Through the equation (9) and $\boldsymbol{P}\boldsymbol{P}^{\top} = \boldsymbol{I}$, we apply the same $\boldsymbol{P}$ to $\boldsymbol{\alpha}^{\gamma\top}$ and rewrite the objective function of problem (7) as:

$$\begin{aligned}\sum_{v=1}^{M} \alpha_v^{\gamma} \mathcal{L}^{v}(\boldsymbol{z}^{v}, y) &= \boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)\boldsymbol{\alpha}^{\gamma} = \boldsymbol{\mathcal{L}}(\boldsymbol{z}, y)\boldsymbol{P}(\boldsymbol{\alpha}^{\gamma\top}\boldsymbol{P})^{\top} \\ &= \sum_{v=1}^{M} \alpha_{\mathcal{P}(v)}^{\gamma} \mathcal{L}^{\mathcal{P}(v)}(\boldsymbol{z}^{\mathcal{P}(v)}, y)\end{aligned} \tag{10}$$

Based on equation (10), we select first s smallest elements and optimize their corresponding weights $\alpha_{\mathcal{P}(v)}|_{v=1}^{s}$, meanwhile, setting the rest $M-s$ weights $\alpha_{\mathcal{P}(v)}|_{v=s+1}^{M}$ as zeros. Therefore, the problem (7) is equivalent to the following problem by absorbing the constraint $\|\boldsymbol{\alpha}\|_0 = s$ into the objective function:

$$\min_{\alpha_{\mathcal{P}(v)} \geq 0,\ \sum_{v=1}^{s} \alpha_{\mathcal{P}(v)} = 1} \sum_{v=1}^{s} \alpha_{\mathcal{P}(v)}^{\gamma} \mathcal{L}^{\mathcal{P}(v)}(\boldsymbol{z}^{\mathcal{P}(v)}, y) \tag{11}$$

Through the Lagrangian Multiplier method, taking the derivatives of $L(\alpha_{\mathcal{P}(v)}, \lambda)$ with respect to $\alpha_{\mathcal{P}(v)}$ and λ, respectively, and setting them to zeros, there is:

$$\alpha_{\mathcal{P}(v)} = \frac{\mathcal{L}^{\mathcal{P}(v)}(\boldsymbol{z}^{\mathcal{P}(v)}, y)^{\frac{1}{1-\gamma}}}{\sum_{w=1}^{s} \mathcal{L}^{\mathcal{P}(w)}(\boldsymbol{z}^{\mathcal{P}(w)}, y)^{\frac{1}{1-\gamma}}}, v = 1, \cdots, s \tag{12}$$

where s is the sparsity of $\boldsymbol{\alpha}$ and $\alpha_{\mathcal{P}(v)} = 0$ if $v = s{+}1, \cdots, M$. According to the property of $\frac{1}{1-\gamma}$ in equation (12), when γ is greater than or equal to a threshold, the weights of all the views will approach $\frac{1}{s}$, which leads to treating all the views equally and is adverse to select some discriminative views.

4 Experiments

In this section, we evaluate the proposed method on the ModelNet40 dataset described in section 4.1 and make comparisons with several state-of-the-art methods.

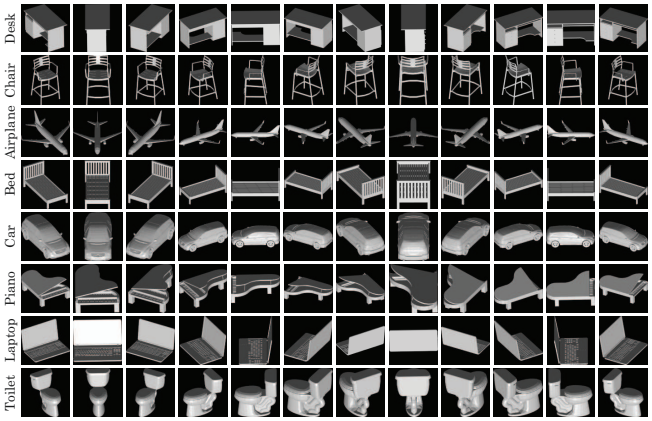

Figure 2: Example views generated by some categories of 3D objects in ModelNet40.

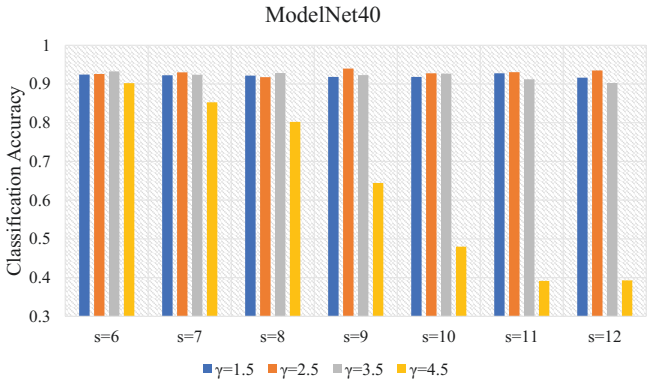

Figure 3: The parameters s and γ in our proposed method on ModelNet40 dataset, where s denotes the number of nonzero elements in $\boldsymbol{\alpha}$ and γ is the power exponent of $\boldsymbol{\alpha}$.

Method	Classification (Accuracy)
MVCNN (VGG-M)	89.90%
MVCNN (ResNet-18)	93.20%
GVCNN (GoogLeNet)	92.60%
GVCNN (ResNet-18)	93.10%
Ours (ResNet-18)	**94.16%**

Table 1: Performance of the proposed method Ours, MVCNN, and GVCNN based on different architectures on the ModelNet40 dataset.

Method	Classification (Accuracy)
Ours (concatenation)	87.92%
Ours ($\boldsymbol{\alpha}$)	93.49%
Ours (s)	93.30%
Ours ($\boldsymbol{\alpha} + s$)	93.58%
Ours ($\boldsymbol{\alpha} + s$+inter)	**94.16%**

Table 2: Ablation study of our proposed method on ModelNet40 dataset.

4.1 Datasets

The classification in 3D is mainly based on the Computer-Aided Design (CAD) model. One widely used dataset is ModelNet [Wu *et al.*, 2014] that has 127915 3D CAD models from 662 categories. ModelNet40 [Wu *et al.*, 2015] provided on the Princeton ModelNet website [1] is a subset of the ModelNet and has 12311 models from 40 common categories. Figure 2 selects 8 kinds of simple categories to intuitively show 12 2D views rendered from a 3D object, where 12 views are generated from 360 degrees with an interval of 30 degrees. For the classification task, all the works are discussed on the ModelNet40, referring to [Su *et al.*, 2015] to conduct the training/testing split.

4.2 Experimental Settings

We compare our proposed method with several state-of-the-art methods for multi-view 3D object classification, including three both view and volume-based methods (MVCNN-MultiRes [Qi *et al.*, 2016], FusionNet [Hegde and Zadeh, 2016], Minto [Minto *et al.*, 2018]), two typical volume-based methods (3DShapeNets [Wu *et al.*, 2014], 3D-A-Nets [Ren *et al.*, 2017]), two typical pointset-based methods (PointNet [Qi *et al.*, 2017], SO-Net [Li *et al.*, 2018]), and three typical view-based methods (Pairwise [Johns *et al.*, 2016], MVCNN [Su *et al.*, 2015], GVCNN [Feng *et al.*, 2018]).

It is worth mentioning that ResNet-18 as the base architecture in our experiments is used to learn high-level information. Considering the FLOPs of the deeper CNNs (e.g., ResNet-50/152) and a better trade-off between accuracy and memory cost compared to other classical CNNs (e.g., VGG, GoogLeNet), ResNet-18 is a good choice but not limited to this CNN architecture. To evaluate the base architectures, we compare the results of MVCNN and GVCNN with ResNet-18, whose results are shown in Table 1. Obviously, the use of ResNet-18 can improve the performance of MVCNN and GVCNN. For example, MVCNN (ResNet-18) with 12 views achieves 3.3% improvements compared with MVCNN (VGG-M). Using the same base architecture, GVCNN (ResNet-18) with 12 views achieves 0.5% gains compared with GVCNN (GoogLeNet) in the classification tasks.

For our proposed method, we fine-tune the parameters of ResNet-18 using the ModelNet40 dataset and use Adam with learning_rate $= 5 * 10^{-6}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, weight_decay $= 0.001$, batch_size $= 8$, epoch $= 30$ for optimization. Furthermore, there are two parameters s and γ in the proposed method, where s denotes the number of nonzero elements in $\boldsymbol{\alpha}$ and γ is the power exponent of each element of $\boldsymbol{\alpha}$. For one thing, we tune s in the range of $[6, 12]$ with step 1 to select a few discriminative and informative views to make a joint decision during classification. For another thing, we vary γ from 1.5 to 10 with a step of 1 to explore the influence on different values of γ on classification accuracy. Based on the proper parameters $s = 9$ and $\gamma = 2.5$, we can train an optimal model to improve the performance of classifying 3D objects significantly. The variations of s and γ in our method on the ModelNet40 dataset are shown in Figure 3.

[1] http://modelnet.cs.princeton.edu/

Method	Input	#View	Classification (Accuracy)
MVCNN-MultiRes [Qi *et al.*, 2016]	view+volume	/	91.40%
FusionNet [Hegde and Zadeh, 2016]	view+volume	/	90.80%
Minto [Minto *et al.*, 2018]	view+volume	/	89.30%
3DShapeNets [Wu *et al.*, 2014]	volume	1	77.00%
3D-A-Nets [Ren *et al.*, 2017]	volume	1	90.50%
PointNet [Qi *et al.*, 2017]	pointset	1	89.20%
SO-Net [Li *et al.*, 2018]	pointset	1	93.40%
Pairwise [Johns *et al.*, 2016]	view	12	90.70%
MVCNN [Su *et al.*, 2015]	view	12	89.90%
GVCNN [Feng *et al.*, 2018]	view	12	92.60%
Ours ($\boldsymbol{\alpha} + s$+inter)	view	12	**94.16%**

Table 3: Comparison of classification accuracy. The proposed method outperforms other state-of-the-art methods on ModelNet40 dataset.

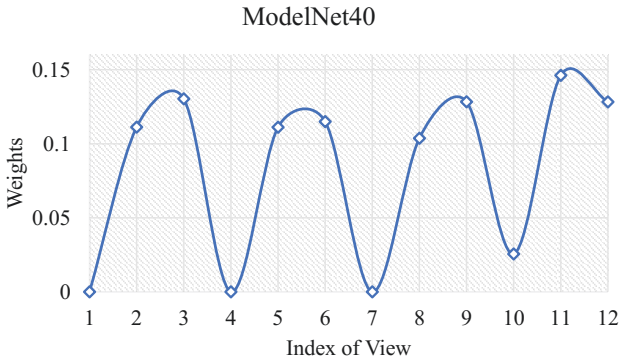

Figure 4: The weights of 12 views learned by our proposed method on ModelNet40 dataset, where the x-axis denotes the index of different views and the y-axis corresponds to the weight of each view.

4.3 Experimental Results

We evaluate the performance of different modules of the proposed method and report the results in Table 2. The ablation studies demonstrate that the weight distribution $\boldsymbol{\alpha}$, the sparsity of multiple views s, and the inter-view information for any different views play different roles during classifying 3D objects. First, all the multi-view methods outperform the single-view method (concatenating all the views as one view and perform the single-view version of our method on it), which verifies the advantages of multi-view representations. Second, the classification accuracy of Ours ($\boldsymbol{\alpha} + s$) is better than that of Ours ($\boldsymbol{\alpha}$) and Ours (s), respectively. It is obvious that considering the weight distribution and the sparsity of multiple views simultaneously is reasonable and effective. Finally, Ours ($\boldsymbol{\alpha} + s$+inter) obtains better performance than any other method, which shows that inter-view information across views also plays an important role.

The experimental results of different methods and their comparisons are reported in Table 3. The proposed method Ours ($\boldsymbol{\alpha} + s$+inter) with 12 views achieves the best classification accuracy. Firstly, compared with the 'view+volume'-based methods, i.e., MVCNN-MultiRes, FusionNet, and Minto, our proposed method gains 2.76%, 3.36%, and 4.86% improvements, respectively. It is obvious that the inputs of these methods contain both 2D and 3D information, however making them work well with each other needs to be improved. Secondly, compared with the volume-based methods, i.e., 3DShapeNets and 3D-A-Nets, our proposed method obtains 17.16% and 3.66% improvements, respectively. It is found that these volume-based methods also cannot address 3D volumetric data processing effectively. Thirdly, making comparisons between the pointset-based methods (including PointNet and SO-Net) and our proposed method, the performance of classifying 3D objects can be achieved 4.96% and 0.76% improvements, respectively. However, the problem of effectively modeling point clouds still needs to be solved. Finally, compared with other view-based methods, such as Pairwise, MVCNN, and GVCNN, our proposed method achieves 3.46%, 4.26%, and 1.56% improvements, respectively. This verifies the superiority of our method at integrating the inter-view information and a selective and adaptive weighting strategy into a unified multi-view framework.

Figure 4 shows the learned weights of different views on ModelNet40 dataset. The higher weight indicates that the view provides more valuable information and makes more contributions during the multi-view 3D object classification.

5 Conclusion

In this paper, we propose a novel 2D-CNNs based multi-view framework for 3D object classification. We take the multiple 2D images rendered from the 3D CAD model as the inputs and develop an end-to-end multi-view framework. It not only integrates the discriminative information with relationships among views but also provides a novel view ensemble mechanism for fusing multiple views to jointly make a decision for classifying 3D objects. The experimental results verify the superiority and effectiveness of our method in 3D object classification.

Acknowledgements

This work was supported in part by the National Key R & D Program under Grant 2017YFB1002201, Innovation Foundation for Doctor Dissertation of NWPU (No. CX201814), State Key Laboratory of Geo-Information Engineering (No. SKLGIE2017-Z-3-2), and Research Funds for Interdisciplinary Subject, NWPU.

References

[Brock *et al.*, 2016] 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.

[Chatfield *et al.*, 2014] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *arXiv preprint arXiv:1405.3531*, 2014.

[Feng *et al.*, 2018] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *CVPR*, pages 264–272, 2018.

[Guo *et al.*, 2013] Yulan Guo, Ferdous Sohel, Mohammed Bennamoun, Min Lu, and Jianwei Wan. Rotational projection statistics for 3d local surface description and object recognition. *IJCV*, 105(1):63–86, 2013.

[Guo *et al.*, 2016] Yulan Guo, Mohammed Bennamoun, Ferdous Sohel, Min Lu, Jianwei Wan, and Ngai Ming Kwok. A comprehensive performance evaluation of 3d local feature descriptors. *IJCV*, 116(1):66–89, 2016.

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

[Hegde and Zadeh, 2016] Vishakh Hegde and Reza Zadeh. Fusionnet: 3d object classification using multiple data representations. *arXiv preprint arXiv:1607.05695*, 2016.

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

[Kanezaki *et al.*, 2018] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *CVPR*, pages 5010–5019, 2018.

[Kazhdan *et al.*, 2003] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *SGP*, pages 156–164, 2003.

[Klokov and Lempitsky, 2017] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *ICCV*, pages 863–872, 2017.

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

[Li *et al.*, 2018] Jiaxin Li, Ben M Chen, and Gim Hee Lee. So-net: Self-organizing network for point cloud analysis. In *CVPR*, pages 9397–9406, 2018.

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

[Minto *et al.*, 2018] Ludovico Minto, Pietro Zanuttigh, and Giampaolo Pagnutti. Deep learning for 3d shape classification based on volumetric density and surface approximation clues. In *VISIGRAPP*, pages 317–324, 2018.

[Paletta and Pinz, 2000] Lucas Paletta and Axel Pinz. Active object recognition by view integration and reinforcement learning. *RAS*, 31(1-2):71–86, 2000.

[Qi *et al.*, 2016] 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.

[Qi *et al.*, 2017] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017.

[Ren *et al.*, 2017] Mengwei Ren, Liang Niu, and Yi Fang. 3d-a-nets: 3d deep dense descriptor for volumetric shapes with adversarial networks. *arXiv preprint arXiv:1711.10108*, 2017.

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

[Su *et al.*, 2015] 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.

[Wang *et al.*, 2019] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. *arXiv preprint arXiv:1906.01592*, 2019.

[Wu *et al.*, 2014] Zhirong Wu, Shuran Song, Aditya Khosla, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets for 2.5 d object recognition and next-best-view prediction. *arXiv preprint arXiv:1406.5670*, 2014.

[Wu *et al.*, 2015] 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.

[Yavartanoo *et al.*, 2018] Mohsen Yavartanoo, Eu Young Kim, and Kyoung Mu Lee. Spnet: Deep 3d object classification and retrieval using stereographic projection. In *ACCV*, pages 691–706, 2018.