PointOfView: A Multi-modal Network for Few-shot 3D Point Cloud Classification Fusing Point and Multi-view Image Features

Huantao Ren, Jiyang Wang, Minmin Yang, Senem Velipasalar
Syracuse University, Electrical Engineering and Computer Science Dept., Syracuse, NY, USA
{hren11, jwang127, myang47, svelipas}@syr.edu [*]

Abstract

Most existing 3D point cloud analysis approaches employ traditional supervised methods, which require large amounts of labeled data, and data annotation is labor-intensive, and costly. On the other hand, although many existing works use either raw 3D point clouds or multiple 2D depth images, their joint use is relatively under-explored. To address these issues, we propose PointOfView, a novel, multi-modal few-shot 3D point cloud classification model, to classify never-before-seen classes with only a few annotated samples. A 2D multi-view learning branch is proposed for processing multiple projection images, and it contains two sub-branches to extract information at individual image level as well as among all six depth images. In addition, we propose a multi-scale 2D pooling layer, which employs various 2D max-pooling and 2D average pooling operations, with different pooling sizes. This allows fusing features at different scales. The second main branch processes raw 3D point clouds by first sorting them, and then using DGCNN to extract features. We perform within-dataset and cross-domain experiments on ModelNel40, ModelNet40-C and ScanobjectNN datasets, and compare with six state-of-the-art baselines. The results show that our approach outperforms all baselines in all experimental settings and achieve the state-of-the-art performance.

1. Introduction

With the rapid development and ever-increasing availability of 3D sensing technology, point cloud analysis has attracted increasing attention from the research community. 3D point cloud data is employed in a wide range of applications, including self-driving cars, unmanned vehicles, and robotics.

In recent years, many supervised learning approaches have been proposed for point cloud analysis tasks, such as point cloud classification, segmentation and completion. Based on the input modality, these approaches can be broadly classified into 3 categories, namely volumetric-based, point-based and multi-view-based methods [14]. Volumetric-based methods first transfer point clouds to a set of voxels, and then use a 3D Convolutional Neural Network (CNN) to extract features. Point-based methods utilize raw point clouds as input. With multi-view-based approaches, each 3D object is represented by multiple depth images, which are obtained by projecting the raw 3D point cloud onto planes from different angles.

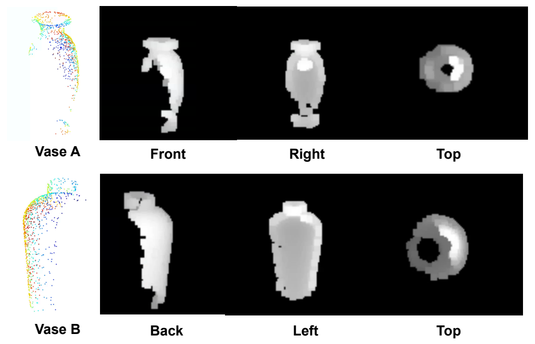

Figure 1. Projection images of different objects of the same class, missing different parts.

However, supervised methods require large amounts of labeled data for training, which limits their applicability, since annotation is a costly and time consuming process. In addition, when faced with unseen classes, these methods cannot provide satisfactory performance. To address these issues, few-shot learning (FSL) has been introduced. FSL allows learning similarities and differences between samples instead of focusing on class-specific features, and thus, allows a network to generalize to novel classes, with only a few annotated samples for each unseen class. In FSL, a sample in the query set is matched to a sample in the support set, which contains a limited number of labeled instances.

Considering this matching nature of the problem, 3D point clouds introduce additional challenges, since they are often affected by occlusion, and suffer from missing points. For instance, in Fig. 1, the top and bottom vases have missing points on their left and right sides, respectively, making the matching of the 3D point clouds more difficult.

[*]This work was supported in part by New York State Department of Economic Development under Contract number C080116.

Yet, some of their 2D projection images, obtained from different view angles, still look similar, which will allow their embeddings to be closer. This example and the further evidence in [2] illustrate the advantages of using multi-view images. Yet, point cloud projection process is influenced by view angles and information is inevitably lost during the process. Employing raw data as input can better preserve the original spatial information and internal structure.

With the pros and cons of multi-view images and 3D point clouds, it is a natural next step to exploit 3D point cloud data and multi-view image data simultaneously to obtain better 3D shape representation. In this paper, we propose PointOfView, a 2-branch backbone for few-shot (FS) point cloud classification. An overview of our approach is presented in Fig. 2. In the 3D point cloud learning branch, we employ DGCNN [25] as the backbone, since it was shown [28] that DGCNN outperforms other backbones in 3D FS point cloud classification. For the 2D multi-view learning branch, we propose a novel, multi-view image processing model with two sub-branches to extract information at individual image level as well as among all six depth images (global information).

In addition to using two modalities, local features are also of great importance for better classification. With FSL, labeled data for each class is limited. Thus, it is essential to extract more informative features from the limited data. Hence, to discover intricate patterns, we propose two local feature extractors, for 3D point cloud and multi-view image processing, to extract more descriptive and distinguishing local features. DGCNN uses EdgeConv to incorporate local neighborhood information. With a similar motivation, we also consider capturing local information to assist final classification. Instead of simply using the points with maximum feature value in each strip/bin, as done in CMFF [27], we propose sorting the point cloud first, after sorting the point cloud along the gravity axis, we divide the points into multiple strips. Then, by applying global max pooling and average pooling in each strip, we can get local features from semantically more meaningful parts of an object. In the 2D multi-view learning branch, to better learn local information, we propose a multi-scale 2D pooling method. ViewNet and CMFF [2, 27] perform global max-pooling on each horizontally split strip, and keep only the highest value features from each strip, which will cause loss of spatial information. In contrast, we employ 3-level 2D max pooling and 2D average pooling ($4 \times 4, 8 \times 8, 16 \times 16$) on the whole feature map, and then aggregate local features from various scales at the end to learn more distinguishing local features.

Main contributions of this work include the following:

- We propose a novel network, PointOfView, for FS 3D point cloud classification, which simultaneously extracts both global and local features from raw 3D point cloud data and visual features from multi-view 2D image data.
- For the 2D multi-view learning branch, we propose an effective structure, composed of two sub-branches, which aggregates individual image-level features and features from groups of images (global information), and incorporates a smoothing layer.
- We propose multi-scale 2D pooling for the 2D multi-view learning branch, which can learn local spatial features at different scales. In the 3D point cloud learning branch, we sort the raw point cloud data along the gravity axis to effectively learn spatial part representations.
- We perform within-dataset and cross-domain experiments, and compare with six state-of-the-art (SOTA) baselines. The results show that our approach outperforms all baselines in all experimental settings, and demonstrates better generalizability.
- We perform detailed ablation studies to further analyze the effectiveness of different components.

2. Related Work

2.1. Point Cloud Classification

As mentioned above, point cloud classification methods can be classified into three broad categories. *Volumetric-based methods* [15, 30] map each point into an occupancy grid, which is sent through 3D CNN networks to perform prediction. *Multi-view projection approaches* render a group of 2D images by projecting points onto planes from different angles. Each image goes through 2D CNN networks to extract view-based features. MVCNN [20] renders snapshots from 12 views, and then processes them independently with a CNN model. Yet, MVCNN does not distinguish between different views, and treats them equally. To address this, GVCNN [4] exploits the relationship between different views (8 or 12) by grouping the set of views based on their discrimination scores. Later on, SimpleView [6] showed that projecting a point cloud onto only 6 (vs. 8 or 12) orthogonal planes, and passing the projection images through ResNet [7] can work well. Recently, Chen et al. [2] proposed a multi-view based backbone for FS 3D point cloud classification by using 6 projection images.

Point-based methods directly employ raw point clouds. PointNet [16] uses shared point-wise MLPs to extract point features, and then applies max-pooling to obtain permutation invariant features. Yet, PointNet does not capture local features. To address this issue, Qi et al. [17] proposed PointNet++, a hierarchical network, to capture local features from the neighborhood of each point. In DGCNN [25], a graph is constructed in the feature space and dynamically updated after each layer, such that neighbors of each point change as the point propagates through the network.

2.2. Few-shot Learning

In general, most existing FSL algorithms follow the meta-learning framework, and can be classified into three cate-

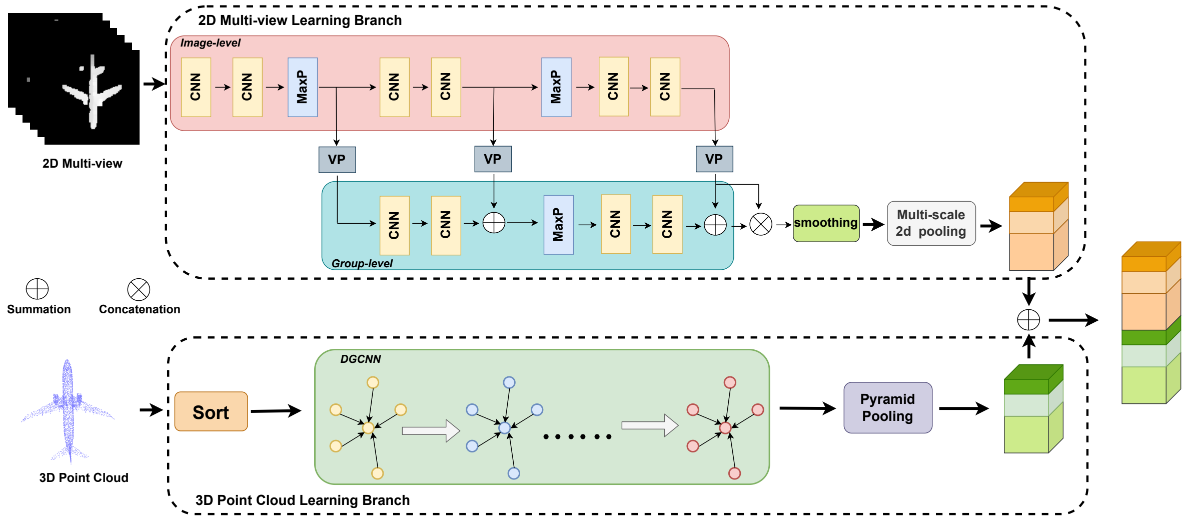

Figure 2. The framework of our proposed method. 'VP', 'smoothing' and 'MaxP' represent view pooling, smoothing layer, and max-pooling operation, respectively. In 2D multi-view branch, image-level sub-branch (pink box) learns features of each projected image, and group-level sub-branch (blue box) learns global features from combinations of six projections. Then, the output features from group-level branch go through multi-scale 2d pooling. In 3D point cloud learning branch, sorted point cloud is sent to DGCNN. Then, pyramid pooling is used to learn part representation. Finally, the features from 2D multi-view and 3D point cloud learning branch are concatenated.

gories: metric-, optimization- and model-based methods.

Metric-based methods strive to preserve class neighborhood structure, by keeping the learned features from the same class closer, and vice versa. Matching Networks [24] use one network for support and one network for query samples, and employ Long Short Term Memory (LSTM) to obtain full context embedding for support samples. Prototypical Networks [19] encode query and support samples into a shared embedding space, and then compute the category prototypes by taking the average of support samples. Query category is determined by using the squared Euclidean distance between a query sample and the prototypes of each category in the feature space.

Optimization-based methods focus on training models that can achieve rapid adaptation. MAML [5], a pioneering work, aims to find a set of initialization parameters such that these learned parameters can achieve the optimal results with a small number of gradient steps.

Model-based methods aim at finding the parameters by using the knowledge learned from different tasks. The goal of FSL is to use the knowledge learned before and learn new information quickly. Hence, some works use memory functions. MANN model [18] modifies the training settings and proposes a new addressing mechanism for assigning attention weights to memory vectors.

2.3. 3D Multi-modal Methods

3D point clouds usually have complex structures, making it hard for a single modality to fully describe a 3D shape. Hence, multi-modal approaches [27, 29] have been proposed in this endeavor. FusionNet [8] jointly employs voxelized data and multi-view images for 3D object classification. MV3D [3] fuses point cloud data from LiDAR and RGB images from camera to perform 3D object detection. Point cloud data is projected into bird's eye view and fusion is performed at image level. Cross-modality feature fusion network (CMFF) [27] fuses 3D point cloud data and 2D projection images for FS 3D point cloud classification. DGCNN and ResNet are used to extract features from point clouds and 2D depth images, respectively. Different from CMFF, we propose multi-scale pooling and sorting methods in 2D multi-view and 3D point cloud branch, respectively, to extract better local features. Moreover, we propose a 2D multi-view feature extractor to learn information at both single-image and group-image levels.

3. Proposed Method

We propose a multi-modal network, PointOfView (POV), for FS 3D point cloud classification, by fusing point cloud data and 2D multi-view depth images. Our goals include (i) exploiting the complementary strengths of these modalities, (ii) extracting more distinguishing local features, and (iii) obtaining a better shape representation from *both* local and global parts, and (iv) achieve better cross-domain generalization thorough combination of global and local features.

3.1. Problem Statement

Let $D = \{P, V\}$ be a given dataset, where $P \in \mathbb{R}^{M\times 3}$ denotes an unordered point set containing M points, with their 3D coordinates, and $V \in \mathbb{R}^{6\times H\times W}$ denotes the rendered 2D image, which is obtained by projecting P onto six orthogonal views [6] as illustrated in Fig. 3. H and W denote the height and width of the projection depth image.

In standard FS classification, given a labeled dataset of base classes C_{base}, the goal is to train a predictor for novel classes C_{novel}, with only few labeled samples, where $C_{base} \cap C_{novel} = \emptyset$. In N-way K-shot Q-query FS classification, the support set $\mathcal{S} = \{P_i^S, V_i^S, y_i^S\}_{i=1}^{N\times K}$ contains K-many labeled samples for each of the N classes. The query set $\mathcal{Q} = \{P_i^Q, V_i^Q, y_i^Q\}_{i=1}^{N\times Q}$ contains samples from the same N classes with Q samples for each class. The goal is to classify $N \times Q$ point clouds into N classes.

3.2. Backbone

The framework of our model, which contains two main branches, is presented in Fig. 2.

The 2D multi-view learning branch, the upper branch in Fig. 2, takes six projection images (left, right, back, front, top and bottom views) as input. We employ a structure composed of two sub-branches. The image-level sub-branch is used to extract features from each projection image independently by a set of convolutional layers and 2D max-pooling operations. Group-level sub-branch learns more global information by using combinations of six projection images. This sub-branch is also composed of convolutional layers and 2D max-pooling. To connect these two sub-branches, and obtain more global and descriptive features from individual image features, we propose an improved View Pooling module [2]. This module allows to aggregate features, obtained from the image-level branch, as shown in Fig. 4. More specifically, six feature maps, from six projection images, are combined in five different ways. Since shapes of objects from opposite projections look similar, and their features are more likely to be closer, we combine features from three opposite pairs ({(left, right), (top, bottom), (front back)}). The remaining combinations are two triplets {(top, front, left), (bottom, back, right)}, each of which and the two combined can describe a 3D shape sufficiently well. Finally, image-level features and group features from 2 sub-branches are added and sent to a smoothing layer, which is a convolutional layer with a kernel size of 3. Smoothing is applied to alleviate the aliasing effect caused by the addition of two feature maps. The effectiveness of the smoothing layer is analyzed in Sec. 5.5. Then, we use

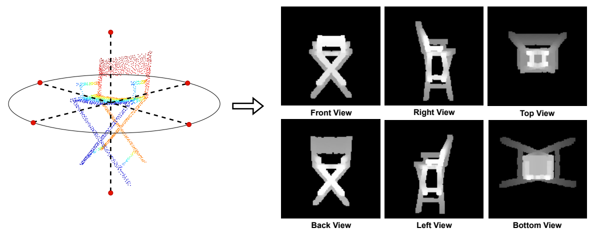

Figure 3. Multi-view images are generated by projecting a point cloud onto six orthogonal planes similar to SimpleView [6].

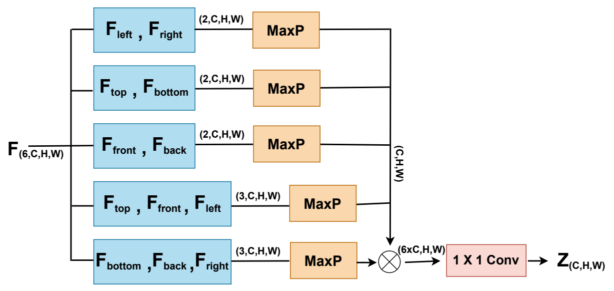

Figure 4. The structure of proposed View Pooling. F and Z are the input and output feature maps, respectively. MaxP represents the max-pooling operation. '$\bigotimes$' denotes concatenation.

our proposed multi-scale 2D pooling to learn features from different receptive fields, details of which are explained in Sec. 3.3.

Our proposed 2D multi-view learning branch is different from the approach in [2] in multiple ways: (i) we apply view pooling and max-pooling across the sub-branches in a different order, i.e. the second interaction between the sub-branches is applied before max-pooling to avoid information loss and provide the group-level branch with more complete feature map; (ii) we do not employ global max-pooling across all views in view pooling, because the adequacy of features derived from the five groups and result is slightly better without global max-pooling; (iii) we only use the feature map from the group-level branch for final prediction, as the local features have been integrated into the group-level branch. (iv) we incorporate a smoothing layer to alleviate the aliasing effect; (v) we propose a multi-scale 2D pooling, described in Sec. 3.3, to learn features from different receptive fields.

The 3D point cloud learning branch, the lower branch in Fig. 2, takes a raw 3D point cloud as input, and first sorts the point cloud along the gravity axis to improve Pyramid Pooling [27], and to make sure that strips used in pyramid pooling correspond to parts of an object. More details will be provided below. Then, we employ DGCNN to extract point features, since Ye et al. [28], who proposed CIA as a FS classification head, have shown that DGCNN provides better results than other point cloud processing models [11–13, 16, 17] on 3D FS classification task. After DGCNN, we adopt Pyramid Pooling [27] to gather both global and local features by splitting the output feature map into strips along the height dimension. Pyramid pooling has six scales such that the feature map is divided into $i \in \{1, 2, 4, 8, 16, 32\}$-many strips. Then, global max pooling and global average pooling are applied among the points in each strip, so that prominent part features are kept, and features are concatenated along the strip dimension. Yet, point clouds are unordered, and these strips do not necessarily represent the corresponding spatial part of an object. To address this, we first sort a point cloud, as mentioned above, so that, e.g. if the object is upright, the first and last strips correspond to the top and bottom parts of an object, respectively, and if the object is upside down, the top part of point cloud is corresponding to the last strip of the feature map. An upright example correspondence is presented in Fig. 5. Finally, the features obtained from the 3D point cloud and 2D depth images are concatenated and sent to the few-shot head. In this work, we employ sqMA [27] as the few-shot head.

3.3. Multi-scale 2D Pooling

The structure of our proposed multi-scale 2D pooling is presented in Fig. 6. The input is the output of the smoothing layer in 2D multi-view processing branch. Then, to learn distinguishing and comprehensive local features, $n \times n$ 2D max-pooling and 2D average pooling operations are performed on spatial dimension to extract features from different scales, where $n \in \{4, 8, 16\}$. Then, outputs of 2D max-pooling and 2D average pooling are added to obtain F' as follows:

$$F' = n \times n \text{ Max-pooling}(F) + n \times n \text{ Avg-pooling}(F). \quad (1)$$

Then, outputs from the three pooling operations are reshaped so that they can be concatenated to fuse local and global features and get the output $F' \in \mathbb{R}^{C \times B}$, where C is the feature dimension, B is number of bins, and $B = (H \times W)/(4 \times 4) + (H \times W)/(8 \times 8) + (H \times W)(16 \times 16)$. The final step is to employ a fully connected layer to get the output O. We also compared the results of exclusively performing $n \times n$ Max-pooling and exclusively performing $n \times n$ Avg-pooling. The analysis is described in Sec. 5.2.

4. Experimental Results

4.1. Training and Testing Details

As mentioned above, we first sort a raw 3D point cloud along the y-axis (gravity direction), and then send the sorted point cloud to DGCNN. For the 2D multi-view learning branch, six input projection images of size 128 × 128 are obtained by projecting the point cloud onto six orthogonal angles. The convolutional channels for the image-level sub-branch and group-level sub-branch are set as (32,32,64,64,128,128) and (64,64,128,128), respectively.

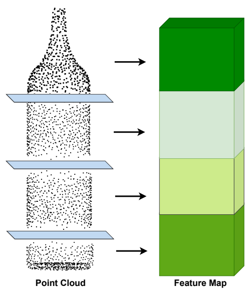

Figure 5. The correspondence between the parts of a sorted point cloud and the strips of a feature map. Only $i = 4$ strips are shown for simplicity.

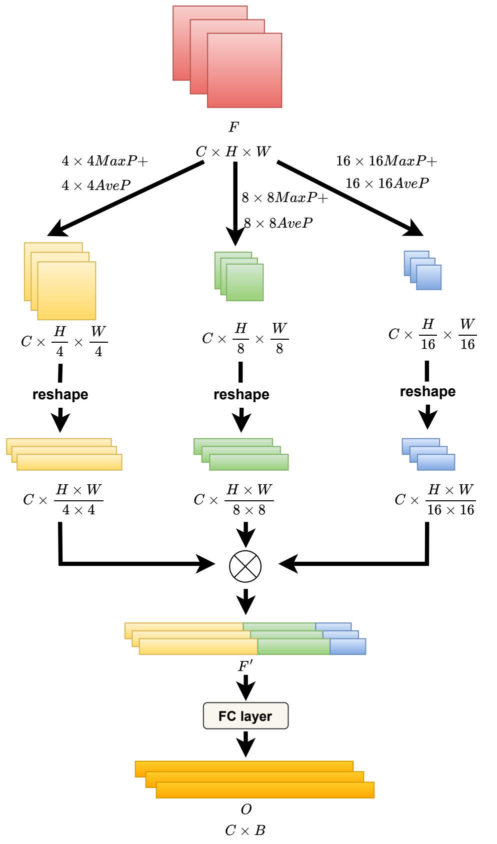
F
$C \times H \times W$
$4 \times 4 MaxP+$
$4 \times 4 AveP$
$8 \times 8 MaxP+$
$8 \times 8 AveP$
$16 \times 16 MaxP+$
$16 \times 16 AveP$
$C \times \frac{H}{4} \times \frac{W}{4}$
$C \times \frac{H}{8} \times \frac{W}{8}$
$C \times \frac{H}{16} \times \frac{W}{16}$
reshape
reshape
reshape
$C \times \frac{H \times W}{4 \times 4}$
$C \times \frac{H \times W}{8 \times 8}$
$C \times \frac{H \times W}{16 \times 16}$
F'
FC layer
O
$C \times B$

Figure 6. The structure of proposed multi-scale 2D pooling. The feature map F is the output of the smoothing layer. C, H and W denote the number of feature channels, and height and width of the feature map, respectively. '$\otimes$' denotes concatenation. F passes through 2D max pooling and average pooling with three different kernel sizes. Then, these feature maps are reshaped and concatenated to get the final feature map F'.

All experiments were performed on one GPU with exactly the same training and testing environment. The optimizer is Adam with an initial learning rate of 0.0008, and gamma is set as 0.5. The learning rate decays every 5 epochs. Following [27], we first meta-train the model for 100 epochs. Each epoch has 400 meta-training episodes. Each episode contains N classes with K labeled samples as support set and Q samples as query set, i.e. it is N-way, K-shot, Q-query setting. During training, we also apply random scaling and translation on points to augment data. After meta-training, we test our model with 700 meta-testing episodes and the same N-way, K-shot, Q-query setting. The final performance is determined by averaging the classification results of these meta-testing episodes with 95% confidence intervals. In this work, Batch All (BA_+) triplet loss [9] is employed to train the network with margin set as 0.2. For multi-scale 2D pyramid pooling, we apply 2D max pooling and 2D average pooling 3 times, with pooling sizes 4, 8, and 16. In addition to within dataset experiments, we perform cross-domain experiments by training on synthetic data, and testing on real-world data.

4.2. Within Dataset Few-shot Learning Results

We first perform within dataset experiments on three different datasets, and compare our PointOfView with six different baselines. We summarize the results in Tables 1, 2 and 3, wherein compared methods are listed as 'Backbone + Few-shot head'. The first four lines in these tables are methods using only 3D point clouds as input with DGCNN as the backbone, and MetaOptNet [10], RelationNet [22], ProtoNet [19] and CIA [28] as the few-shot heads, respectively. DGCNN is used as the backbone, since it was shown in [28] that DGCNN has the best performance on 3D FS classification task. The fifth line (ViewNet+CIA) is another single-modality method using only 2D projection images. CMFF+sqMA [27] is a fusion method using both modalities. To have a commensurate comparison with ViewNet (a sinle-modality method), we also test using our depth image processing branch only with CIA as the few-shot head, and refer to this as 'Our view branch + CIA'.

4.2.1 Within Dataset Results on ModelNet40

ModelNet40 [26] is a synthetic and commonly used dataset, which contains 12,311 CAD-generated meshes from 40 classes. For each 3D model, 1024 points are uniformly sampled from the mesh, and each point has its (x, y, z) coordinates. An example object and six projection images obtained from the point cloud are presented in Fig. 3. To be consistent with the experimental settings of the SOTA baselines [2, 27], we conduct 4-fold cross validation experiments. We sort 40 classes based on their object IDs, and divide them into 4 groups, with 10 classes in each group. We compare our model with six baselines, and summarize the results in Tab. 1. As can be seen, our method outperforms *all* the baselines, including single- and multi-modality approaches, in terms of average accuracy for both 1-shot and 5-shot settings. Even when we only use the depth image processing branch of POV with the CIA head, we still outperform all single-modality approaches (first five rows).

4.2.2 Within Dataset Results on ModelNet40-C

ModelNet40-C [21] is a more recent dataset that has the same number of classes as ModelNet40. Different from ModelNet40, it introduces 15 corruptions, such as LiDAR, occlusion, etc., to simulate real-world point cloud data. Example point clouds and three corresponding projected images are shown in Fig. 7(b). We split this dataset the same way as ModelNet40. The experiment results in Tab. 2 show that our method outperforms *all* baselines for all folds, and for both 1-shot and 5-shot settings. Our depth image processing branch followed by the CIA head also outperforms ViewNet+CIA head for both 1-shot and 5-shot settings. The improvement margins over single modality models range between 1.64% and 11.23% for 1-shot setting, and between

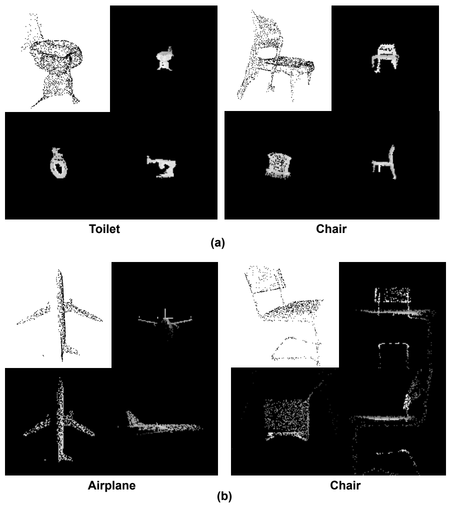

Figure 7. (a) Example point clouds and three projected images from ScanobjectNN [23] and (b) ModelNet40-C [21] datasets.

1.08% and 6.45% for 5-shot setting. POV also surpasses CMFF+sqMA by 2.25% for 1-shot setting, and by 1.47% for 5-shot setting. It should be noted that *the improvement margins on this dataset are higher* than the ones for ModelNet40, which is encouraging and shows that our model can better handle issues, such as missing points, and is a better choice for challenging scenarios.

4.2.3 Within Dataset Results on ScanObjectNN

ScanObjectNN dataset [23] contains 15k objects, with 1024 points, from 15 classes. Different from the above two datasets, ScanObjectNN is collected by scanning real-world objects. Due to occlusions and noise, point clouds are not as perfect as the ones in ModelNet40 and ModelNet40-C, causing this benchmark to be more challenging. Example point cloud objects and the corresponding three projected images are shown in Fig. 7(a). We perform 3-fold cross validation on this dataset by using 5 classes per fold. The results are summarized in Tab. 3. Our method outperforms *all* baselines in all folds, in terms of average accuracy, for both 1-shot and 5-shot settings. It improves the other multi-modal method CMFF+sqMA by 3.24% for 1-shot setting, and by 2.4% for 5-shot setting. The improvement margins over single modality models range between 4.7% and 16.23% for 1-shot setting and between 5.24% and 13.28% for 5-shot setting. Only with our depth image processing branch and the CIA head, our method still outperforms all single-modality methods. *The improvement margins are highest on this dataset*. This, once again, shows that our model is more robust and effective in the challenging, close-to-real-world and real-world situations.

4.3. Cross-domain Few-shot Learning Results

We perform cross-domain experiments to demonstrate the generalization ability of our model. For source-domain datasets, we use ModelNet40, ModelNet40-C

	5-way 1-shot					5-way 5-shot				
	fold0	fold1	fold2	fold3	Avg	fold0	fold1	fold2	fold3	Avg
DGCNN+MetaOptNet [10]	82.87±0.72	75.77±0.83	65.31±0.92	66.97±0.93	72.73±0.85	92.37±0.38	86.44±0.62	82.10±0.58	83.15±0.55	86.02±0.53
DGCNN+RelationNet [22]	82.14±0.69	77.46±0.80	66.09±0.91	69.47±0.84	75.23±0.81	91.53±0.38	85.11±0.61	79.36±0.63	83.01±0.52	84.75±0.53
DGCNN+ProtoNet [19]	85.42±0.64	79.46±0.76	70.06±0.39	70.73±0.42	76.42±0.55	93.99±0.29	88.65±0.54	84.76±0.51	85.56±0.48	88.24±0.45
DGCNN+CIA [28]	89.97±0.63	83.89±0.75	75.31±0.82	79.27±0.77	82.21±0.72	94.61±0.30	89.15±0.50	85.00±0.51	86.71±0.50	88.87±0.47
ViewNet+CIA [2]	92.57±0.52	82.68±0.80	75.28±0.90	80.95±0.75	82.87±0.74	96.23±0.26	89.64±0.55	85.74±0.51	90.18±0.45	90.45±0.44
CMFF+sqMA [27]	92.94±0.47	85.52±0.73	**77.76±0.82**	81.80±0.71	84.50±0.68	96.82±0.22	91.76±0.53	**87.88±0.48**	91.03±0.40	91.85±0.41
Our view branch + CIA	92.18±0.48	85.14±0.70	74.71±0.82	82.17±0.75	83.55±0.0.68	96.81±0.22	91.07±0.49	86.20±0.52	90.96±0.39	91.26±0.41
POV + sqMA (Ours)	**93.44±0.42**	**86.02±0.71**	76.85±0.81	**82.85±0.69**	**84.79±0.66**	**97.30±0.19**	**92.02±0.50**	87.74±0.50	**91.61±0.40**	**92.17±0.40**

Table 1. Few-shot 3D point cloud classification results on the ModelNet40 dataset. **Bold** indicates the best results.

	5-way 1-shot					5-way 5-shot				
	fold0	fold1	fold2	fold3	Avg	fold0	fold1	fold2	fold3	Avg
DGCNN+MetaOptNet [10]	78.28±0.79	75.34±0.84	58.07±0.86	66.29±0.91	69.50±0.85	91.09±0.40	84.19±0.57	75.10±0.73	81.34±0.53	82.93±0.56
DGCNN+RelationNet [22]	79.59±0.74	74.63±0.84	59.03±0.81	68.38±0.86	70.41±0.81	87.12±0.46	83.55±0.54	70.18±0.78	79.01±0.58	79.97±0.59
DGCNN+ProtoNet [19]	81.29±0.71	75.83±0.79	61.76±0.84	69.83±0.84	72.18±0.80	90.97±0.39	86.21±0.50	76.99±0.65	83.19±0.51	84.34±0.51
DGCNN+CIA [28]	85.70±0.75	76.97±0.90	65.68±1.00	74.32±0.94	76.34±0.89	92.07±0.36	86.81±0.56	76.11±0.71	83.71±0.51	84.68±0.54
ViewNet+CIA [2]	89.47±0.58	81.05±0.78	69.56±0.86	76.29±0.85	79.09±0.78	94.95±0.31	88.75±0.49	81.53±0.60	86.78±0.46	88.00±0.47
CMFF+sqMA [27]	88.50±0.59	80.95±0.74	69.81±0.86	74.64±0.82	78.48±0.75	95.11±0.29	89.32±0.46	81.63±0.63	85.58±0.48	87.91±0.47
Our view branch+CIA	89.21±0.59	81.80±0.71	70.61±0.91	78.91±0.78	80.13±0.75	96.13±0.25	89.41±0.48	81.68±0.66	88.55±0.42	88.94±0.45
POV+sqMA (Ours)	**89.77±0.51**	**82.61±0.74**	**72.00±0.88**	**78.52±0.77**	**80.73±0.73**	**96.14±0.25**	**90.11±0.45**	**82.91±0.64**	**88.37±0.42**	**89.38±0.44**

Table 2. Few-shot 3D point cloud classification results on the ModelNet40-C dataset. **Bold** indicates the best results.

	5-way 1-shot				5-way 5-shot			
	fold0	fold1	fold2	Avg	fold0	fold1	fold2	Avg
DGCNN+MetaOptNet [10]	41.92±0.72	61.12±0.66	53.87±0.78	52.30±0.72	63.86±0.56	67.73±0.45	70.19±0.49	67.26±0.50
DGCNN+RelationNet [22]	50.29±0.76	54.23±0.63	51.45±0.64	51.99±0.68	58.65±0.53	66.72±0.50	65.94±0.52	63.77±0.52
DGCNN+ProtoNet [19]	50.81±0.73	60.46±0.67	58.72±0.78	56.66±0.73	68.45±0.54	70.20±0.52	68.76±0.49	69.13±0.52
DGCNN+CIA [28]	50.58±0.82	62.17±0.68	62.59±0.74	58.45±0.75	62.94±0.51	71.31±0.45	70.21±0.48	68.15±0.48
ViewNet+CIA [2]	60.90±0.76	66.48±0.60	64.10±0.77	63.83±0.71	73.66±0.48	74.77±0.45	77.46±0.46	75.30±0.46
CMFF+sqMA [27]	61.09±0.72	66.29±0.65	68.39±0.68	65.29±0.68	74.90±0.48	76.51±0.40	83.02±0.41	78.14±0.43
Our view branch+CIA	61.20±0.68	66.95±0.62	66.49±0.71	64.88±0.67	73.42±0.49	76.05±0.40	78.07±0.46	75.85±0.45
POV+sqMA (Ours)	**64.03±0.67**	**70.04±0.59**	**71.53±0.69**	**68.53±0.65**	**80.21±0.42**	**76.75±0.40**	**84.67±0.39**	**80.54±0.40**

Table 3. Few-shot 3D point cloud classification results on the ScanObjectNN dataset. **Bold** indicates the best results.

	ShapeNet-XFS $\rightarrow$ ScanObjectNN		ModelNet40-XFS $\rightarrow$ ScanObjectNN		ModelNet40-C-XFS $\rightarrow$ ScanObjectNN	
	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot	5-way 1-shot	5-way 5-shot
DGCNN+ProtoNet	44.39±0.8	62.12±0.72	53.04±0.91	65.53±0.75	53.7±0.89	65.81±0.76
DGCNN+CIA	49.29±0.9	64.37±0.77	55.59±1.01	65.61±0.74	57.07±0.94	66.29±0.76
ViewNet+CIA	53.38±0.88	69.46±0.74	60.35±0.93	75.38±0.74	61.28±0.98	74.48±0.71
CMFF+sqMA	52.98±0.78	69.75±0.69	61.45±0.87	75.23±0.65	61.76±0.86	74.38±0.65
POV+sqMA (Ours)	**53.71±0.77**	**72.38±0.65**	**65.37±0.84**	**78.58±0.62**	**63.46±0.86**	**77.88±0.62**

Table 4. The cross-domain few-shot classification results. **Bold** indicates the best results.

and ShapeNetCore [1], since they contain 3D synthetic CAD models. We use the real-world dataset ScanObjectNN, with all 15 classes, as the target-domain dataset. To avoid having overlapping classes between source and target datasets, we construct ModelNet40-XFS, ModelNet40-C-XFS and ShapeNetCore-XFS from ModelNet40, ModelNet40-C and ShapeNetCore, respectively. Both ModelNet40-XFS and ModelNet40-C-XFS contain 26 base classes and ShapeNetCore-XFS includes 44 base classes. We perform three cross-domain FSL experiments, ModelNet40-XFS→ScanObjectNN, ModelNet40-C-XFS→ScanObjectNN and ShapeNetCore-XFS→ScanObjectNN. The training procedure is the same as within dataset experiments. Here, we have chosen the better performing baselines from the previous experiments to compare with. The results in Tab. 4 show that our model outperforms all baselines for all experiments, highlighting its better generalizability across domains. It is interesting to see that in 1-shot setting, our method achieves higher accuracy on ModelNet40-XFS→ScanObjectNN than ModelNet40-C-XFS→ScanObjectNN, which is in contrast to baselines. Moreover, the improvement margin, compared to baselines, is higher for 1-shot setting than 5-shot setting. These show that our method can extract more meaningful and transferable features with limited data.

5. Ablation Studies

To verify the effectiveness of different components of our proposed method, ablation studies are carried out on the ScanObjectNN dataset, since it is the most challenging one among the three datasets.

5.1. The impact of sorting point clouds

As explained above, we sort a point cloud along the gravity axis before sending it to DGCNN. Tab. 5 shows the accuracy of our model with and without sorting raw point clouds. As can be seen, with sorting, the accuracy of the model improves by 1.97% and 1.93% in 1-shot and 5-shot settings, respectively.

		fold0	fold1	fold2	Avg
5-way 1-shot	w/o sorting	60.89%	67.58%	71.21%	66.56%
	w/ sorting	**64.03%** (**↑3.14%**)	**70.04%** (**↑2.46%**)	**71.53%** (**↑0.32%**)	**68.53%** (**↑1.97%**)
5-way 5-shot	w/o sorting	75.49%	75.42%	**84.93%**	78.61%
	w/ sorting	**80.21%** (**↑4.72%**)	**76.75%** (**↑1.33%**)	84.67% (↓0.26%)	**80.54%** (**↑1.93%**)

Table 5. The accuracy of models with and without sorting the raw point cloud in the beginning.

5.2. The impact of multi-scale 2D pooling

To verify the effectiveness of our proposed multi-scale 2D pooling, we replace it with the Pyramid Pooling [27] and then compare the performances. Table 6 shows that (red vs. green rows for 1-shot and 5-shot settings) our multi-scale 2D pooling provides 0.84% and 0.33% improvement in 1-shot and 5-shot settings, respectively.

Our approach uses a combination of 2D max pooling and 2D average pooling. Tab. 6 also shows a comparison when each of these pooling operations is used by themselves (blue rows). Our approach, using the combination, outperforms applying either operation alone in most folds. We believe the reason is that combining 2D max pooling and 2D average pooling can help to extract important features and obtain more generalized representation of the local regions.

	pyramid pooling	2D max pooling	2D avg pooling	fold0	fold1	fold2	Avg
5-way 1-shot	✓			63.61±0.72	69.14±0.59	70.32±0.71	67.69±0.67
		✓		64.69±0.68	68.84±0.59	71.34±0.72	68.29±0.66
			✓	**65.07±0.63**	69.32±0.58	70.41±0.73	68.26±0.65
		✓	✓	64.03±0.67	**70.04±0.59**	**71.53±0.69**	**68.53±0.65**
5-way 5-shot	✓			80.01±0.45	76.72±0.43	83.89±0.40	80.21±0.43
		✓		80.05±0.44	77.36±0.4	**82.48±0.43**	79.96±0.42
			✓	79.98±0.46	**77.38±0.42**	81.45±0.41	79.6±0.43
		✓	✓	**80.21±0.42**	76.75±0.40	84.67±0.39	**80.54±0.40**

Table 6. The impact of different components of multi-scale 2D Pooling and the comparison with pyramid pooling [27].

5.3. Analysis of the number of views

Results of using different number of views (6,8,10 or 14) to train our model are illustrated in Fig. 8. In the case of 6 views, we use the 6 orthogonal views. For 8 views, we select the projections from the vertices of the cube centered on the object. For 10 views, we select the first 8 views around the object every 45 degrees. Remaining two views are top and bottom. For 14 views, we combine the 6 orthognal views and 8 views. The results illustrate that 6 orthogonal views provide comparable if not better accuracy. Although for 5-shot setting, the performance with 14 views is slightly better (1% higher) than using 6 views, processing 14 views demands more computing power. Using 6 views conserves computational resources while effectively capturing features of an object.

5.4. Analysis of computational complexity

We compare the computational complexity of our method with the second best performing model, CMFF, which is also a two-modality model. The comparison, presented in Tab. 7, shows that the total size of our model is less than

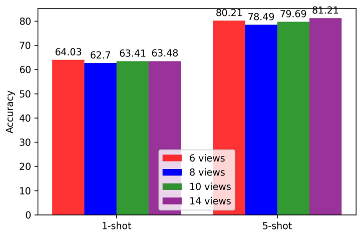

Figure 8. The comparison of using different number of views.

CMFF and our model provides better accuracy and lower standard deviation at the same time.

Model	Forward/backward pass size (MB)	Params Size (MB)	Estimated Total Size (MB)
CMFF	7746.23	6.44	7753.04
POV (Ours)	**5730.39**	**6.97**	**5737.72**

Table 7. Network complexity of CMFF and our proposed POV.

5.5. The impact of the smoothing layer

We proposed to use a smoothing layer, which is a convolutional layer with a kernel size of 3, in the 2D multi-view processing branch to alleviate the aliasing effect caused by the semantic gap between two sub-branches and the summation of two sub-branches. To verify its effectiveness, we compare the performance of the model with and without the smoothing layer. The results, summarized in Tab. 8, show that smoothing layer provides 0.27% and 0.74% improvement in 1-shot and 5-shot settings, respectively.

		fold0	fold1	fold2	Ave
5-way 1-shot	w/o smooth Lyr	63.72%	69.44%	**71.62%**	68.26%
	w/ smooth Lyr	**64.03%(↑0.31%)**	**70.04%(↑0.6%)**	71.53%(↓0.09%)	**68.53%(↑0.27%)**
5-way 5-shot	w/o smooth Lyr	79.97%	**76.91%**	82.53%	79.80%
	w/ smooth Lyr	**80.21%(↑0.24%)**	76.75%(↓0.16%)	**84.67%(↑2.14%)**	**80.54%(↑0.74%)**

Table 8. The model accuracy with and without smoothing layer.

6. Conclusion

We have proposed a novel, multi-modal network, with more generalizability across domains, by combining features from raw 3D point cloud data and multiple 2D depth images for few-shot 3D point cloud classification. To generate more descriptive and distinguishing features from six projected images, we have proposed a 2D Multi-view Learning Branch to extract information both at image- and group-level. The group-level features allow capturing more global information about the shape. We have also proposed a multi-scale 2D pooling approach, which allows extracting features from different scales. We have performed comprehensive set of experiments within datasets and cross-domains for 5-way 1-shot and 5-way 5-shot 3D point cloud classification. Results have shown that our method outperforms six baselines on all three benchmark datasets, and improvement margin is higher on more challenging datasets, namely ScanobjectNN and ModelNet40-C. This shows the great potential of our method for dealing with challenging cases, such as missing points and occlusion. In cross-domain experiments, our model shows better generalization ability compared to four baselines.

References

[1] 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. 7

[2] Jiajing Chen, Minmin Yang, and Senem Velipasalar. Viewnet: A novel projection-based backbone with view pooling for few-shot point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17652–17660, 2023. 2, 4, 6, 7

[3] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 3

[4] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 264–272, 2018. 2

[5] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017. 3

[6] Ankit Goyal, Hei Law, Bowei Liu, Alejandro Newell, and Jia Deng. Revisiting point cloud shape classification with a simple and effective baseline. In *International Conference on Machine Learning*, pages 3809–3820. PMLR, 2021. 2, 4

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

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

[9] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 5

[10] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10657–10665, 2019. 6, 7

[11] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems*, 31, 2018. 4

[12] Yongcheng Liu, Bin Fan, Gaofeng Meng, Jiwen Lu, Shiming Xiang, and Chunhong Pan. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5239–5248, 2019.

[13] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8895–8904, 2019. 4

[14] Haoming Lu and Humphrey Shi. Deep learning for 3d point cloud understanding: a survey. *arXiv preprint arXiv:2009.08920*, 2020. 1

[15] Daniel Maturana and Sebastian Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 922–928. IEEE, 2015. 2

[16] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017. 2, 4

[17] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017. 2, 4

[18] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016. 3

[19] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 3, 6, 7

[20] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[21] Jiachen Sun, Qingzhao Zhang, Bhavya Kailkhura, Zhiding Yu, Chaowei Xiao, and Z Morley Mao. Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*, 2022. 6

[22] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018. 6, 7

[23] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1588–1597, 2019. 6

[24] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 3

[25] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 2

[26] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 6

[27] Minmin Yang, Jiajing Chen, and Senem Velipasalar. Cross-modality feature fusion network for few-shot 3d point cloud classification. In *Proceedings of the IEEE/CVF Winter Con-*

ference on Applications of Computer Vision, pages 653–662, 2023. 2, 3, 4, 5, 6, 7, 8

[28] Chuangguan Ye, Hongyuan Zhu, Yongbin Liao, Yanggang Zhang, Tao Chen, and Jiayuan Fan. What makes for effective few-shot point cloud classification? In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1829–1838, 2022. 2, 4, 6, 7

[29] Haoxuan You, Yifan Feng, Xibin Zhao, Changqing Zou, Rongrong Ji, and Yue Gao. Pvrnet: Point-view relation neural network for 3d shape recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 9119–9126, 2019. 3

[30] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 2