Cross-Modality Feature Fusion Network for Few-Shot 3D Point Cloud Classification

Minmin Yang*, Jiajing Chen*, Senem Velipasalar
Electrical Engineering and Computer Science Dept., Syracuse University, Syracuse, NY, USA
{myang47, jchen152, svelipas}@syr.edu

Abstract

Recent years have witnessed significant progress in the field of few-shot image classification while few-shot 3D point cloud classification still remains under-explored. Real-world 3D point cloud data often suffers from occlusions, noise and deformation, which make the few-shot 3D point cloud classification even more challenging. In this paper, we propose a cross-modality feature fusion network, for few-shot 3D point cloud classification, which aims to recognize an object given only a few labeled samples, and provides better performance even with point cloud data with missing points. More specifically, we train two models in parallel. One is a projection-based model with ResNet-18 as the backbone and the other one is a point-based model with a DGCNN backbone. Moreover, we design a Support-Query Mutual Attention (sqMA) module to fully exploit the correlation between support and query features. Extensive experiments on three datasets, namely ModelNet40, ModelNet40-C and ScanObjectNN, show the effectiveness of our method, and its robustness to missing points. Our proposed method outperforms different state-of-the-art baselines on all datasets. The margin of improvement is even larger on the ScanObjectNN dataset, which is collected from real-world scenes and is more challenging with objects having missing points.

1. Introduction

Point cloud classification is a fundamental task in 3D computer vision, and plays a vital role in different applications including autonomous vehicles, robotics, etc. With the success of the deep learning-based methods in 2D computer vision tasks, more attention has been paid to deep learning-based point cloud analysis [28, 29, 39]. However, supervised deep learning methods rely on large numbers of well-annotated training data. Even if large amounts of 3D point cloud data can be collected by scanning devices, such as Lidar or depth cameras, data annotation is labor-intensive and time-consuming. To address this issue, few-shot learning (FSL) trains a network with strong generalization capability so that it can perform prediction on objects, which are not seen during the training, with only a few labeled samples.

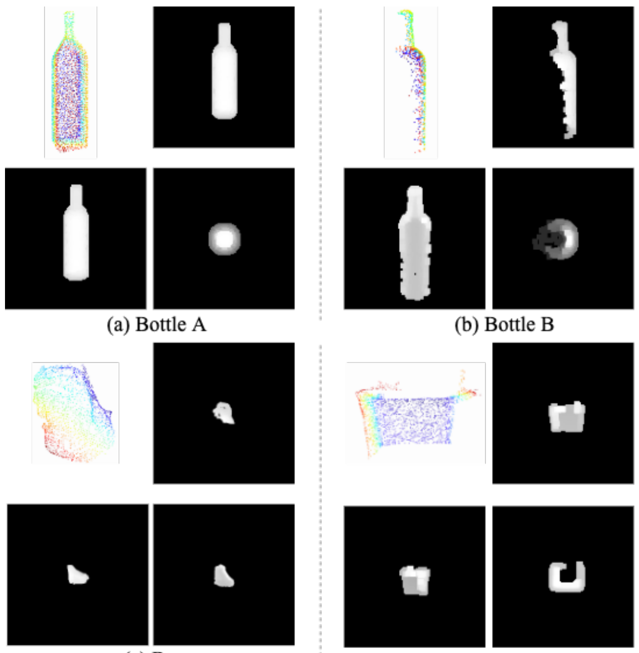


Figure 1. Example point clouds and three of the corresponding depth images obtained by projecting point clouds from multiple orthogonal views [12] for objects from different datasets: (a) a bottle from the ModelNet40 dataset (sampled from a CAD model, no occlusion); (b) a bottle with missing points from the ModelNet40-C dataset; (c) a bag from the ScanObjectNN dataset; (d) a box from the ScanObjectNN dataset. As shown in (b), one or more of the depth images can well compensate for missing point issue. On the other hand, as seen in (c) and (d), point clouds can describe objects with more details than depth images. Some parts, e.g. the bag straps, might be deemphasized due to projection to depth images.

Different from 2D images, 3D point clouds are un-

*Both authors contributed equally. This work was supported in part by the National Science Foundation under Grant 1816732.

structured. Thus, traditional convolutional neural networks (CNNs) cannot be readily applied to 3D point cloud data. Moreover, as mentioned by Ye et al. [42], existing 3D point cloud datasets [41, 38, 35] contain smaller amounts of data compared to 2D image datasets [7, 23], affecting the learning ability of supervised models.

On the other hand, 3D point clouds obtained by scanning real-world objects are often affected by occlusions, and have missing points. Thus, the network should be robust against these issues. Fig. 1 shows example point clouds and three of the corresponding depth images, obtained from different views, for objects from three different datasets. As illustrated in Fig. 1 (b), although points of the bottle are partially missing, the shape and structure of the bottle can still be well described by one of the side-view depth images. On the other hand, depth images do not always depict shapes of objects in as much detail as point clouds as shown in Fig. 1 (c) and Fig. 1 (d). In such cases, point clouds can make up for it and provide more shape information and details.

Considering the complementary properties of the two modalities (point clouds and depth images) of 3D data, we propose a cross-modality feature fusion network, which combines these two input modes and processes them with two different backbones to extract feature maps F_r and F_d from depth images and point cloud data, respectively. The depth images are processed by ResNet-18 [14], and the raw point cloud data is processed by DGCNN [39]. Instead of only using F_r or F_d, our experiments show that a better few-shot classification accuracy can be achieved by fusing F_r and F_d. The reason is that although point-based methods can output a feature map accurately describing the shape and spatial properties of objects, when there are missing points, these features are negatively effected. Since not all depth images are greatly affected, incorporating information from them provides robustness against missing points.

Moreover, to retain more information for further learning, we replace max/average pooling operations in DGCNN and ResNet, since according to [4], a great number of points are discarded during the max pooling operations. Instead, we follow the idea of Horizontal Pyramid Mapping [3] and adapt it to design a pyramid pooling operation for our model, wherein a feature map is split into strips of different scales along the last feature dimension. Then, global average pooling operation and global max pooling operation are applied to each strip to gather global and local information.

In addition, we propose an attention module, referred to as the Support-Query Mutual Attention (sqMA), to update support and query features. Different from the Cross-Instance Fusion (CIF) presented in [42], our sqMA module takes the support features as inputs, instead of prototype features, so that it can take all the support features into consideration to update the query features. Another difference between our sqMA and CIF is that the sqMA module takes all support (query) features to adaptively update query (supports) features, while CIF uses fixed values of K1 most similar query features and K2 most similar prototype features. The main contributions of this work include the following:

1. We analyze the individual drawbacks of point cloud data and depth images, and propose a network, referred to as the Cross-Modality Feature Fusion Network, to process and fuse features from projected depth images and raw point cloud data.
2. We propose a novel Support-Query Mutual Attention (sqMA) module that can update query and support features mutually based on their correlation.
3. To replace traditional average/max pooling, we design a pyramid pooling operation, which can retain more features globally and locally.
4. Our proposed method outperforms several baselines on ModelNet40-C, ModelNet40 and ScanObjectNN datasets by large margins, especially on the ScanObjectNN dataset that contains data collected from real-world scenes, affected with missing points.

The code link is provided in the supplementary material.

2. Related Work

2.1. 3D Point Cloud Classification

The success of deep learning in 2D image processing has driven the development of approaches for deep learning-based 3D point cloud classification [2, 39, 24, 29, 12]. Based on the representation of 3D data, as a structured grid or raw points, existing works can be broadly divided into two classes: point-based and projection-based.

Point-based methods [6, 24] process point clouds directly. PointNet [28] extracts point-wise features by a stack of multi-layer perceptrons and applies max-pooling operations to obtain permutation invariant features. However, PointNet does not encode local structures explicitly. PointNet++ [29] applies PointNet recursively on a partition of the point set and learns local features by taking metric space distances into account. Hence, PointNet++ captures finer-grained details and is more robust than PointNet. Wang et al. [39] propose EdgeConv, which captures local geometric information among points and is invariant to permutations.

Projection-based methods render 3D points to other structured representations, e.g. multi-view projection [13, 34] or voxels [30]. Multi-view based methods are arising due to the success of CNNs in image processing. MVCNN [34] renders 2D images from 12 different views. The images are sent independently to a CNN to get several independent descriptors, which are aggregated by a view pooling layer and then sent to another CNN for classification. Multiple views do not contribute equally for classification, so they should have different weights. To exploit the correlation and distinguishability among views, Feng et al. [10] proposed a group-view CNN that contains a hierarchical view-group-shape architecture. To obtain a good performance, some works [40, 18] require a large number of views. On the contrary, SimpleView [12] does not rely on any modules for view pooling or feature selection and does not depend on pre-trained networks. SimpleView renders depth images by projecting points onto six orthogonal planes. These depth images are fed into ResNet-18 to extract features that are then fused for classification.

2.2. Few-shot Meta-learning

Few-shot learning can quickly adapt to new tasks given a few labeled examples. Meta-learning aims to learn meta-knowledge from many similar tasks by a meta-learner, which has good generalization ability. Therefore, meta-learning has been widely used to solve the few-shot classification problem [36, 16, 5, 27]. Based on meta-learning framework, few-shot learning can be broadly categorized into three classes: optimization-based, model-based and metric-based methods.

Optimization-based methods. The main idea behind optimization-based methods is to differentiate the optimization process over support-set based on the meta-learning framework. The objective of MAML [11] is to find a good parameter initialization so that the model can perform well given a new task by taking several gradient steps. Many variants of MAML have been proposed. Jamal et al. [16] introduced a task-agnostic meta-learning method that prevents the meta-learner from over-performing on certain tasks. Fallah et al. [8] proposed HF-MAML that resolves the complexity bounds of MAML without Hessian-vector product computation. The MetaOptNet [19] incorporates a differentiable quadratic programming solver that can equip the embedding model with varied linear classifiers.

Model-based methods. Different from optimization-based methods, which aim to optimize quickly, model-based methods tailor the model architectures for fast learning. A common way of modifying model architectures is to use external memory. The memory serves as a buffer that networks can use to store new data and retrieve old data. Santoro et al. [31] designed a memory-augmented neural network to quickly assimilate new data. MM-Net [1] writes support features into the memory and reads from the memory at the inference stage. Besides, a contextual learner is proposed to predict the parameters of CNNs that are used for extracting query features.

Metric-based methods. These methods aim to learn a feature representation with a metric. Prototypical Network [32] computes prototypes for each category by taking the mean of support set and classifies query images by calculating the squared Euclidean distance between prototypes and query features. RelationNet [37] incorporates a learnable metric module to determine if query examples and support examples are from the same categories or not. In this work, we propose a metric-based few-shot network with sqMA and pyramid pooling modules to learn representative embeddings and use a nearest-neighbor classifier to recognize query examples.

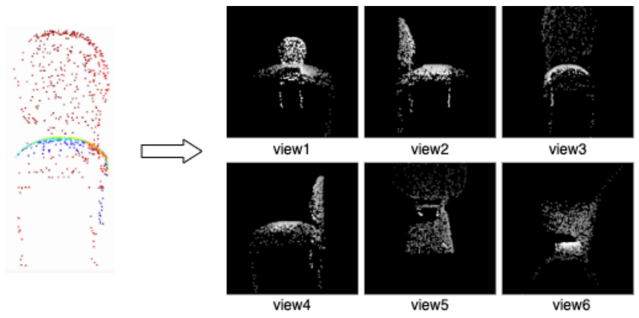


Figure 2. Depth images are generated by projecting a point cloud onto six orthogonal planes as described in SimpleView [12].

2.3. Deep Multi-modal Learning

Multi-modal learning relates information from different modalities to boost a network's performance. Multi-modal data is more informative than uni-modal data, since it can represent the same object with various modalities, which are typically complimentary. Many works on 3D object detection exploit the complementary feature of multi-modal networks and achieve good performance. Liang et al. [22] employed continuous convolutions to fuse features from image and LiDAR at multiple scales. However, 2D-3D constrains between images and LiDAR are ignored. Zhu et al. [43] proposed a multi-modal fusion network that takes images and point clouds as input. The model contains point-wise feature fusion at first stage and RoI-wise deep feature fusion at the second stage. They also designed a 2D-3D coupling loss to constrain 3D detection with 2D detection.

In zero-shot learning, different approaches [20, 17, 9] have been presented to align representations from various data modalities, commonly language and visual. Considering that 3D object shape plays a dominant role in classification, Stojanov et al. [33] incorporate shape bias by using point clouds to learn a discriminative embedding space. The learned embedding space is then used to map images into it. Different from [33], our work does not try to minimize the distance between the images and the corresponding point cloud embeddings but simply concatenates embeddings for further learning.

3. Proposed Method

3.1. Problem definition

Let $\mathcal{P} = \{p_1, p_2, p_3, \cdots, p_n\}$ be a set of 3D points, where $p_i = (x_i, y_i, z_i)$. Following SimpleView [12], we

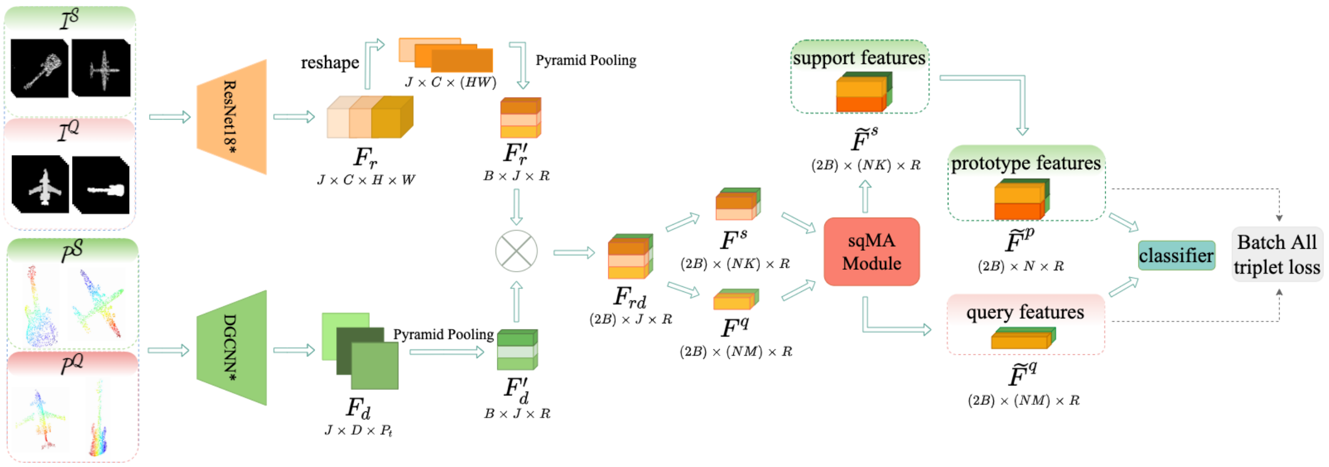


Figure 3. The proposed model architecture for few-shot 3D point cloud classification. Let $\mathcal{P}^S$ and $\mathcal{P}^Q$ denote a point cloud support set and query set, respectively, and let $\mathcal{I}^S$ and $\mathcal{I}^Q$ represent the corresponding depth image support set, and depth image query set, respectively. '$\bigotimes$' is the concatenation operation. Features extracted from ResNet-18* and DGCNN* are denoted by F_r and F_d, respectively. The * indicates that we only adopt part of these networks. After the pyramid pooling operation, two sets of features, $F_r^{'}$ and $F_d^{'}$, are concatenated and sent to the sqMA module to update the support features and query features through mutual guidance. We set Euclidean distance metric as classifier. Finally, triplet loss is computed between prototype features and query features, and is used for updating the model through back-propagation. For better visualization, we only present 2-way 1-shot 1-query setting.

generate depth images from point cloud data by projecting points onto six orthogonal planes, as shown in Fig. 2. For p_i, whose coordinates are with respect to the camera position, the corresponding pixel $\hat{p_i}'$ in the depth image is obtained by projecting point p_i to 2D coordinates at depth z_i, i.e. $\hat{p_i}' = (\hat{x}_i = x_i/z_i, \hat{y}_i = y_i/z_i)$. Then, the 2D coordinates are discretized, $\hat{p}_i = (\lceil \hat{x}_i \rceil, \lceil \hat{y}_i \rceil)$. Each depth image $\mathcal{V}_i \in \mathbb{R}^{H \times W}$, and the set of six depth images is denoted as $\mathcal{I} = \{\mathcal{V}_1, \mathcal{V}_2, \cdots, \mathcal{V}_6\} \in \mathbb{R}^{6 \times H \times W}$.

In our N-way K-shot M-query FSL setting, a support set $\mathcal{S} = \{(\mathcal{P}_i^S, \mathcal{I}_i^S, \mathcal{Y}_i^S)\}_{i=1}^{N \times K}$ includes N classes with K labeled examples for each class. A query set $\mathcal{Q} = \{\mathcal{P}_i^Q, \mathcal{I}_i^Q, \mathcal{Y}_i^Q)\}_{i=1}^{N \times M}$ contains the same N categories as the support set with M testing examples for each category. $\mathcal{P}_i^S$ ($\mathcal{P}_i^Q$) denotes the point cloud data, $\mathcal{I}_i^S$ ($\mathcal{I}_i^Q$) is the set of depth images projected from $\mathcal{P}_i^S$ ($\mathcal{P}_i^Q$) and $\mathcal{Y}_i^S$ ($\mathcal{Y}_i^Q$) is the label. We adopt a meta-learning strategy with a collection of meta-training tasks, defined as $\mathcal{T}_r = \{(\mathcal{S}_i, \mathcal{Q}_i)\}_{i=1}^{T}$ and a set of meta-testing tasks, $\mathcal{T}_t = \{(\mathcal{S}_i, \mathcal{Q}_i)\}_{i=1}^{V}$. Thus, $(J = N \times K + N \times M)$ total examples are given in each meta-training/meta-testing task. In consistence with standard FSL, classes included in $\mathcal{T}_r$ are called base classes, $\mathcal{C}^{base}$, classes in $\mathcal{T}_t$ are called novel classes, $\mathcal{C}^{novel}$, and $\mathcal{C}^{novel} \cap \mathcal{C}^{base} = \emptyset$. The objective of meta-learning algorithms is to learn a good embedding model. Formally,

$$\phi^* = \underset{\phi}{\mathrm{argmin}} \mathbb{E}_{\mathcal{T}_r}[\mathcal{L}(\mathcal{Q}; \phi)], \tag{1}$$

where $\mathcal{L}$ is the loss function, and ϕ is the parameters for embedding networks.

3.2. Network Architecture

As discussed in Sec. 1, point cloud data can have more detailed description of a 3D object's shape, while depth images can be more robust to missing points and deformation. Thus, these two data modalities can complement each other, and their combination can provide more information for recognition. To this end, we propose a cross-modality feature fusion network to fully employ both raw point cloud and projected depth image data, and introduce a simple way to fuse features from these two modalities. Ye et al. [42] have studied the influence of backbones on FSL, and shown that DGCNN provides the best performance as a point cloud backbone compared to other methods [28, 29, 21, 26, 25]. Thus, we adopt DGCNN as the backbone to process raw point cloud data. After using SimpleView to generate depth images from point cloud data, as mentioned in Sec. 3.1, we employ ResNet-18 as the 2D image processing backbone. Moreover, we replace the traditional average/max pooling at the end of the original ResNet-18 and DGCNN, and design a pyramid pooling operation instead, which can retain more features globally and locally.

The overall architecture of our approach is presented in Fig. 3. In the upper branch of the network, depth images in the support set, $\mathcal{I}^S$, and the query set, $\mathcal{I}^Q$, are processed by ResNet-18. In the bottom branch, the point cloud data in the support set, $\mathcal{P}^S$, and the query set, $\mathcal{P}^Q$, are fed into the DGCNN. After this, depth image features F_r and point cloud features F_d are obtained, where $F_r \in \mathbb{R}^{J \times C \times H \times W}$ and $F_d \in \mathbb{R}^{J \times D \times P_t}$. J represents the total number of support and query samples in each episode, C is the number of channels, and H and W are the height and width of feature map F_r, respectively. As for the shape of F_d, D denotes the feature dimension of each point and P_t is the number of points in each sample. After obtaining depth image and point cloud features, we adopt a pyramid pooling operation to collect both global and local features while discarding redundant features. The details of the pyramid pooling are explained in Sec. 3.3.

After pyramid pooling, the support sample's feature F_r' and query sample's feature F_d' are concatenated, and then sent to our proposed sqMA module. In the sqMA module, the discrepancy between the support and query features from the same class can be reduced by computing their correlation. After that, we follow the idea of Prototypical Network [32] to determine the class of query samples. We first compute the prototype features for each class by taking the mean of the updated support features for each class and denote them as $\widetilde{F}^p$. Then, the class distribution of a query example is computed based on softmax output over the distance between the query example and prototypes.

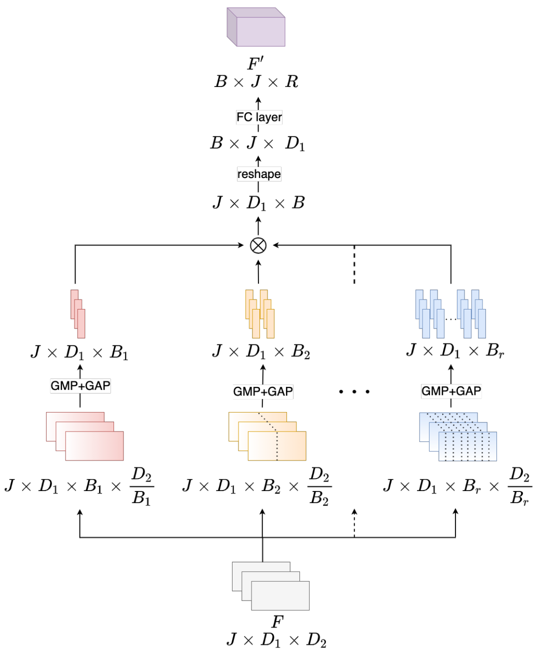
$$p(\mathcal{Y}_i = c|\widetilde{F}_i^q) = \frac{exp(-dist(\widetilde{F}_i^q, \widetilde{F}_c^p))}{\sum_{j=1}^{N} exp(-dist(\widetilde{F}_i^q, \widetilde{F}_j^p))}, \tag{2}$$

where $dist(\cdot,\cdot)$ is the Euclidean Distance, c stands for the class c. Finally, Batch All (BA_+) triplet loss [15] is adopted for loss calculation.

3.3. Pyramid Pooling

The structure of the pyramid pooling operation, which is inspired by Horizontal Pyramid Mapping (HPM) [3], is shown in Fig. 4. Pyramid pooling takes point or image features as input, where J is the number of samples, D_1 is the feature dimension, and D_2 is the number of points in each point cloud or spatial size of depth image depending on the input modality. Next, the input feature map F is divided into B_i bins, with each bin containing $\frac{D_2}{B_i}$-many points or pixels. Then, global max pooling and average pooling are performed along the spatial or point number dimension to obtain a bin's feature matrix. As the value of B_i varies, the receptive field for each bin's features also differs. Thus, by concatenating features of all bins in the end, features from different receptive fields can be covered.

As shown in Fig. 3, pyramid pooling is applied on both depth image feature F_r and point cloud feature F_d. For F_r, we reshape the spatial dimension and split it into strips. In other words, the pyramid pooling operation works on spatial features and prominent features will be kept. For F_d, pyramid pooling is applied to point features dimension so that significant points are conserved. Finally, we apply a fully-connected layer to project features to the same high dimensional space so that they can be concatenated for later operations.



F'
$B \times J \times R$
FC layer
$B \times J \times D_1$
reshape
$J \times D_1 \times B$
$J \times D_1 \times B_1$
$J \times D_1 \times B_2$
$J \times D_1 \times B_r$
GMP+GAP
GMP+GAP
GMP+GAP
$J \times D_1 \times B_1 \times \frac{D_2}{B_1}$
$J \times D_1 \times B_2 \times \frac{D_2}{B_2}$
$J \times D_1 \times B_r \times \frac{D_2}{B_r}$
F
$J \times D_1 \times D_2$

Figure 4. The structure of pyramid pooling. GMP and GAP denote global max pooling and global average pooling, respectively. FC is a fully-connected layer. The feature map F is split into strips of different sizes on last dimension for r times. B is the total number of strips, i.e. $B = \sum_{i=1}^{r} B_i$. Then, global max pooling and global average pooling are applied to each strip to extract global features and local features. These features are concatenated, reshaped and sent to a FC layer to get the final feature map $F' \in \mathbb{R}^{B\times J\times R}$.

3.4. Support-Query Mutual Attention Module

Even though point cloud data can provide more detailed information about object shapes than 2D images, instances belonging to the same class can still have large variations especially when data is partially occluded or corrupted with a cluttered background. In other words, every sample in the support set has different correlation with query examples and makes a different contribution when classifying query examples. Therefore, we design the Support-Query Mutual Attention module that takes the correlation between support features and query features to adaptively update query features and support features, as shown in Fig. 5. To update the support features $F^s \in \mathbb{R}^{2B\times(NK)\times R}$ by query features, $F^q \in \mathbb{R}^{2B\times(NM)\times R}$, we first compute the cosine similarity, $CS(F_{b,i}^s, F_{b,j}^q)$ as:

$$CS(F_{b,i}^s, F_{b,j}^q) = \frac{F_{b,i}^s F_{b,j}^{qT}}{\|F_{b,i}^s\| \cdot \|F_{b,j}^q\|}, \tag{3}$$

where $F_{b,i}^s \in \mathbb{R}^{1\times R}$ and $F_{b,j}^q \in \mathbb{R}^{1\times R}$.

Then, softmax operation is performed for each support feature to normalize its pairwise correlation:

$$W_{b,i,j}^{s2q} = \frac{exp(CS(F_{b,i}^s, F_{b,j}^q))}{\sum_{l=1}^{NM} exp(CS(F_{b,i}^s, F_{b,l}^q))}. \tag{4}$$

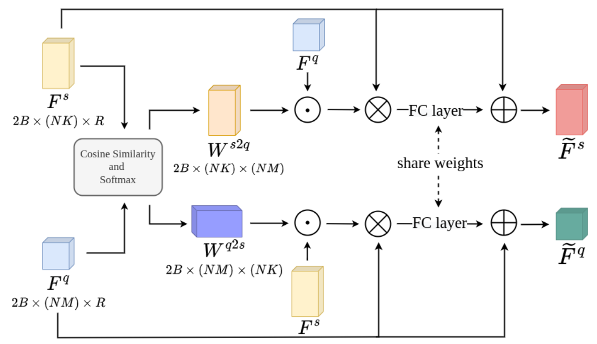

F^s
$2B \times (NK) \times R$
Cosine Similarity and Softmax
F^q
$2B \times (NM) \times R$
W^{s2q}
$2B \times (NK) \times (NM)$
W^{q2s}
$2B \times (NM) \times (NK)$
F^q
F^s
FC layer
share weights
FC layer
$\widetilde{F}^s$
$\widetilde{F}^q$

Figure 5. The sqMA architecture. '$\odot$', '$\bigoplus$' and '$\bigotimes$' denote matrix multiplication, sum and concatenation, respectively.

Based on W^{s2q}, we perform matrix multiplication between $W^{s2q} \in \mathbb{R}^{2B\times(NK)\times(NM)}$ and query features to get the support-weighted query features, denoted as F^{s2q}, i.e. $F^{s2q} = W^{s2q}F^q$. The higher the similarity between the i^{th} support example and the j^{th} query example, the higher the value of $W^{s2q}_{b,i,j}$ is. Thus, in F^{s2q}, query features that are more similar to support features will be highlighted. Then, we concatenate F^{s2q} with support features to enrich the support features with similar query features and apply a fully-connected layer and a ReLU layer to fuse the features and make the sqMA module learn adaptive features for different support-query correlation.

$$F_{s\_cat\_q} = Concat([F^s, F^{s2q}]), \tag{5}$$

$$F^{'}_{s\_cat\_q} = ReLU(FC(F_{s\_cat\_q})). \tag{6}$$

Finally, we add F^s to $F^{'}_{s\_cat\_q}$ to obtain the updated support features $\widetilde{F}^s$ and ease the learning process by skipping some operations in sqMA.

$$\widetilde{F}^s = F^s + F^{'}_{s\_cat\_q}. \tag{7}$$

Then, we take the mean of the updated support features to get the prototype representation of each class, which is denoted by $\widetilde{F}^p$.

Different from CIF [42], our sqMA module is applied to support features, and we compute prototype features based on the updated support features for two reasons. First, the discrepancy between prototype features and query features from the same class might be large considering that only a few examples are provided. Prototypes that have large variance to query features may not be valuable to update query features. Secondly, the average operation often causes information loss, and we want to retain more information for query set to learn the correlation.

The same process can be applied to enhance query features by support features. We can compute W^{q2s} and multiply W^{q2s} with F^s to get the query-weighted support features, F^{q2s}. F^{q2s} is concatenated with F^q and a fully connected layer and ReLU are applied in order to get $F^{'}_{q\_cat\_s}$. The updated query features, $\widetilde{F}^q$ are computed by adding $F^{'}_{q\_cat\_s}$ to F^q.

$$\widetilde{F}^q = F^q + F^{'}_{q\_cat\_s}. \tag{8}$$

As illustrated in Fig. 5, two fully-connected layers share weights to better formulate the correlation between query features and support features.

4. Experiments

CIA [42], with DGCNN as the backbone, has been shown to achieve SOTA performance on few-shot point cloud classification task. Thus, we use CIA as one of the baselines for comparison. In addition, we also compare with three other 2D image few-shot heads, namely MetaOptNet [19], RelationNet [37] and ProtoNet [32], by using DGCNN as their backbone. DGCNN was chosen as the backbone, since the experiments in [42] have shown that DGCNN, as a point cloud processing backbone, provides better performance compared to other methods [28, 29, 21, 26, 25]. For a fair comparison, we also use DGCNN as backbone with only point cloud data as input to show the effectiveness of our few-shot learning mechanism, and denote it as 'Ours*' in Tables 1, 2 and 3. We perform experiments on three point cloud datasets, namely ModelNet40 [41], ModelNet40-C [35] and ScanObjectNN [38] with n-fold cross validation.

4.1. Backbone Architectures

Similar to [12], we use ResNet-18 to extract features from $6 \times 128 \times 128$ depth images. The network starts with a convolution layer, with 3×3 kernel, followed by batch normalization and ReLU operation. Then, there are 3 residual blocks. The feature maps obtained after 3 residual blocks are used for further learning. DGCNN consists of four EdgeConv blocks. The feature maps from each EdgeConv block are concatenated and sent to a fully-connected layer to aggregate features. Then, the feature maps go through pyramid pooling. The feature map F_r has the shape of $J \times 128 \times 32 \times 32$ and F_d has the shape of $J \times 1024 \times 1024$.

4.2. Datasets and Settings

ModelNet40 [41] contains 12,308 CAD models from 40 classes. These CAD models are complete and clean. Each point cloud sample, containing 1024 points, is uniformly sampled from the CAD models and normalized into a unit sphere. For few-shot classification, we conduct 4-fold cross validation for better evaluation. More specifically, we split the dataset into 4 groups based on class IDs, where each group contains 10 categories.

	5-way 1-shot						5-way 5-shot				
	fold 0	fold 1	fold 2	fold 3	Average		fold0	fold 1	fold 2	fold 3	Average
MetaOptNet	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
RelationNet	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
ProtoNet	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
CIA	89.97±0.63	83.46±0.83	74.08±0.95	76.13±0.86	80.91±0.82		94.61±0.30	89.15±0.50	85.00±0.51	86.71±0.50	88.87±0.47
Ours*	<u>90.36±0.54</u>	<u>83.89±0.75</u>	<u>75.31±0.82</u>	<u>79.27±0.77</u>	<u>82.21±0.72</u>		<u>95.71±0.26</u>	<u>90.64±0.52</u>	<u>87.17±0.49</u>	<u>90.51±0.41</u>	<u>91.01±0.42</u>
Ours	**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**	**87.78±0.48**	**91.03±0.40**	**91.85±0.41**

Table 1. Few-shot classification results on the ModelNet40 dataset. **Bold** and <u>underline</u> show the best and the second best result, respectively

	5-way 1-shot						5-way 5-shot				
	fold 0	fold 1	fold 2	fold 3	Average		fold 0	fold 1	fold 2	fold 3	Average
MetaOptNet	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
RelationNet	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
ProtoNet	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
CIA	<u>85.70±0.75</u>	79.67±0.90	65.68±1.00	<u>74.32±0.94</u>	76.34±0.89		92.07±0.36	86.81±0.56	76.11±0.71	83.71±0.51	84.68±0.54
Ours*	85.31±0.67	<u>80.01±0.80</u>	<u>68.79±0.86</u>	73.70±0.85	<u>76.95±0.80</u>		<u>92.72±0.36</u>	<u>88.11±0.49</u>	<u>80.90±0.64</u>	<u>84.40±0.49</u>	<u>86.53±0.50</u>
Ours	**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**

Table 2. Few-shot classification results on the ModelNet40-C dataset. **Bold** and <u>underline</u> show the best and the second best result, resp.

	5-way 1-shot					5-way 5-shot			
	fold 0	fold 1	fold 2	Average		fold 0	fold 1	fold 2	Average
MetaOptNet	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
RelationNet	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
ProtoNet	50.81±0.73	60.46±0.67	58.72±0.78	56.66±0.73		68.42±0.54	70.20±0.52	68.76±0.49	69.13±0.52
CIA	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
Ours*	<u>58.76±0.76</u>	<u>64.69±0.64</u>	<u>67.47±0.73</u>	<u>63.64±0.71</u>		<u>72.09±0.50</u>	<u>74.60±0.43</u>	<u>78.92±0.42</u>	<u>75.20±0.45</u>
Ours	**61.09±0.72**	**66.29±0.65**	**68.39±0.68**	**65.26±0.68**		**74.90±0.48**	**76.51±0.40**	**83.02±0.41**	**78.14±0.43**

Table 3. Few-shot classification results on the ScanObjectNN dataset. **Bold** and <u>underline</u> show the best and the second best result, resp.

ModelNet40-C [35] has the same number of classes as ModelNet40, but the data is meticulously corrupted with 15 realistic corruptions to simulate real-world scenarios. The same 4-fold cross validation setup as ModelNet40 is used.

ScanObjectNN [38], different from the above two datasets, is a real-world point cloud dataset, including scans of indoor scenes. It has 15 classes and 2,902 examples. With this dataset, a 3-fold cross validation is performed with 5 classes per fold.

We believe that experiments conducted on the ScanObjectNN and ModelNet40-C datasets better demonstrate the potential of our method for real-world applications, since they contain data affected by missing points.

4.3. Implementation Details

All experiments use Adam optimizer with an initial learning rate of 8×10^{-4} and gamma of 0.5. The learning rate decays every 5 epochs. Following [42], the model is meta-trained for 100 epochs, and each epoch contains 400 episodes. During meta-training, we apply random rotation and jitter to augment data. After meta-training, meta-testing with 700 episodes is conducted. Similar to the meta-training phase, meta-testing phase also uses DGCNN and ResNet-18 as feature extractors. The reported accuracy is the average over meta-testing episodes with 95% confidence intervals. At the meta-training stage, we employ BA_+ triplet loss and set the margin to 0.2. In our implementation, for pyramid pooling operation, the total number of strips, B, is 63. We split the feature map 6 times, with 1, 2, 4, 8, 16 and 32 strips, respectively. The feature maps $F_r^{'}$ and $F_d^{'}$ have the shape of $63 \times J \times 256$.

4.4. Discussion of Results

We conduct 5-way 1-shot 10-query and 5-way 5-shot 10-query classification. The results on ModelNet40, ModelNet40-C and ScanObjectNN datasets are shown in Tables 1, 2 and 3, respectively, wherein 'Ours*' denotes our model using only point cloud data, and 'Ours' denotes our full model using point clouds and depth images. As can be seen, our full method outperforms all baselines on all three datasets for average accuracy across folds as well as all individual folds. The margin of improvement is especially larger, as much as 9.99%, for 5-way 5-shot classification compared to the second best performer, on the more challenging ScanObjectNN dataset.

As for the baselines, RelationNet [37] incorporates a relation module to directly sum up all support features from the same class, and its performance can be affected by high intra-class differences of support examples. ProtoNet [32] takes the mean of support features for each class, and performs better than RelationNet. ProtoNet is comparable to CIA on ModelNet40 and ModelNet40-C datasets and even surpasses CIA on the ScanObjectNN dataset in 5-shot setting. However, in 1-shot setting, CIA obtains higher average accuracy than ProtoNet. This can be attributed to the cross-instance adaption module designed for addressing problems of subtle inter-class differences and high intra-class variances. When comparing the results of 'Ours*' to other baselines, as seen from Tables 1, 2 and 3, 'Ours*' (using only point cloud data) still outperforms all baselines on all datasets in terms of average accuracy. For instance, CIA provides average accuracy of 58.45% and 68.15% in 1-shot and 5-shot settings, respectively, while Ours* achieves 63.64% and 75.20% accuracy in 1-shot and 5-shot settings, respectively. This shows that our method can still provide good performance by only utilizing point features from DGCNN, and that the improvement is due to our proposed pyramid pooling and sqMA modules. The pyramid pooling module is proposed to retain more useful information and the sqMA module is designed to utilize the correlation among examples. By introducing two data modalities, our method (denoted by Ours) provides even higher accuracy, which demonstrates the effectiveness of fusing two data modalities.

4.5. Ablation Studies

We conduct ablation studies on three aspects to verify the effectiveness of our method: (1) we compare our method, which uses both depth image data and point cloud data, with using depth images only or point clouds only; (2) we analyze the effectiveness of our sqMA module by testing our model with and without sqMA, and by also incorporating the proposed sqMA in a different baseline model; (3) we compare the performances of using pyramid pooling and using global max pooling, and show that global max pooling discards a lot of useful information, and pyramid pooling operation can rectify this by retaining more features globally and locally. The ablation studies are conducted on the ScanObjectNN dataset based on 700 meta-testing episodes in 5-way 1-shot setting and 5-way 5-shot setting.

Effects of Combining Two Data Modalities. Results of using different data modalities as input are shown in Tab. 4. Using depth images and point data together provides consistently better performance than using single data modality in 1-shot and 5-shot settings. The results strongly support our initial observation that two modalities are complementary and their combination allows the network to learn distinguishing features. Moreover, we visualized query feature embeddings from different models in Fig. 6. As can be seen, our model fusing two data modalities (Fig. 6 (c)) can learn more distinguishing features, and the class boundaries are more precise and compact.

	Input Modality	fold 0	fold 1	fold 2	Average
5-way 1-shot	Depth images only	50.06	65.48	58.63	58.06
	Point cloud only	58.76	64.69	67.47	63.64
	Point cloud+Depth images	**61.09**	**66.29**	**68.39**	**65.26**
5-way 5-shot	Depth images only	65.08	73.57	68.57	69.07
	Point cloud only	72.09	74.60	78.92	75.20
	Point cloud+Depth images	**74.90**	**76.51**	**83.02**	**78.14**

Table 4. The accuracy of models with different input modalities.

Effect of the sqMA Module. We proposed the sqMA module to update support and query embeddings by considering their mutual correlation. Tab. 5 shows the accuracy of our model w/ and w/o the sqMA module. With the sqMA module, the model reaches higher accuracy, with 2.07% increase in 1-shot setting and 1.45% increase in 5-shot setting. We also used the sqMA module in MetaOptNet [19] and showed the improvement in the suppl. material.

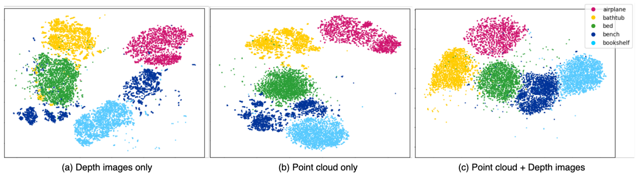


Figure 6. The t-SNE visualization of query feature distributions when different data modalities are used as input.

		fold 0	fold 1	fold 2	Average
5-way 1-shot	w/o sqMA	58.18	64.53	66.87	63.19
	w/ sqMA	**61.09**	**66.29**	**68.39**	**65.26**
5-way 5-shot	w/o sqMA	72.84	75.93	81.29	76.69
	w/ sqMA	**74.90**	**76.51**	**83.02**	**78.14**

Table 5. The accuracy of models with and without sqMA module.

Effect of Pyramid Pooling. To verify the effectiveness of pyramid pooling, we replace it with a max-pooling operation. Tab. 6 shows that pyramid pooling provides 2.65% accuracy improvement in 1-shot setting and 3.3% improvement in 5-shot setting. Given that the accuracy is improved more with the 5-shot setting than 1-shot, it can be concluded that max-pooling operation causes loss of useful information, especially when more labeled examples are provided, and the pyramid pooling can alleviate this issue.

		fold 0	fold 1	fold 2	Average
5-way 1-shot	max pooling	56.86	64.24	66.73	62.61
	pyramid pooling	**61.09**	**66.29**	**68.39**	**65.26**
5-way 5-shot	max pooling	72.02	73.18	79.32	74.84
	pyramid pooling	**74.90**	**76.51**	**83.02**	**78.14**

Table 6. The accuracy of models using pyramid pooling operation and max-pooling operation.

5. Conclusion

We have proposed a cross-modality few-shot feature learning network to learn and fuse features from depth images and point cloud data for 3D point cloud classification. This approach exploits the complementary aspects of these modalities. In addition, to address the issue of large intra-class variations of samples in a support set, we have presented a Support-Query Mutual Attention (sqMA) module, which updates support/query features by computing the similarity between them. We have conducted extensive experiments and ablation studies and shown that our method outperforms different baselines on three different datasets, and improvement margin is even higher on the more challenging ScanOjbectNN dataset. We have also shown the effectiveness of the proposed sqMA module.

References

[1] Qi Cai, Yingwei Pan, Ting Yao, Chenggang Yan, and Tao Mei. Memory matching networks for one-shot image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4080–4088, 2018.

[2] Weiwei Cai, Dong Liu, Xin Ning, Chen Wang, and Guojie Xie. Voxel-based three-view hybrid parallel network for 3d object classification. *Displays*, 69:102076, 2021.

[3] Hanqing Chao, Yiwei He, Junping Zhang, and Jianfeng Feng. Gaitset: Regarding gait as a set for cross-view gait recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8126–8133, 2019.

[4] Jiajing Chen, Burak Kakillioglu, Huantao Ren, and Senem Velipasalar. Why discard if you can recycle?: A recycling max pooling module for 3d point cloud analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 559–567, 2022.

[5] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. 2020.

[6] Silin Cheng, Xiwu Chen, Xinwei He, Zhe Liu, and Xiang Bai. Pra-net: Point relation-aware network for 3d point cloud analysis. *IEEE Transactions on Image Processing*, 30:4436–4448, 2021.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[8] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. On the convergence theory of gradient-based model-agnostic meta-learning algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 1082–1092. PMLR, 2020.

[9] Rafael Felix, Ian Reid, Gustavo Carneiro, et al. Multi-modal cycle-consistent generalized zero-shot learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 21–37, 2018.

[10] 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.

[11] 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.

[12] 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.

[13] David Griffiths and Jan Boehm. A review on deep learning techniques for 3d sensed data classification. *Remote Sensing*, 11(12):1499, 2019.

[14] 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.

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

[16] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11719–11727, 2019.

[17] Huajie Jiang, Ruiping Wang, Shiguang Shan, and Xilin Chen. Transferable contrastive network for generalized zero-shot learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9765–9774, 2019.

[18] Asako Kanezaki, Yasuyuki Matsushita, and Yoshifumi Nishida. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5010–5019, 2018.

[19] 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.

[20] Jingjing Li, Mengmeng Jing, Ke Lu, Zhengming Ding, Lei Zhu, and Zi Huang. Leveraging the invariant side of generative zero-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7402–7411, 2019.

[21] 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.

[22] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 641–656, 2018.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] Hongbin Liu, Jinyuan Jia, and Neil Zhenqiang Gong. Pointguard: Provably robust 3d point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6186–6195, 2021.

[25] 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.

[26] 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.

[27] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International Conference on Machine Learning*, pages 2554–2563. PMLR, 2017.

[28] 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.

[29] 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.

[30] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. Classification of point cloud scenes with multiscale voxel deep network. *arXiv preprint arXiv:1804.03583*, 2018.

[31] 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.

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

[33] Stefan Stojanov, Anh Thai, and James M Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1798–1808, 2021.

[34] 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.

[35] Jiachen Sun, Qingzhao Zhang, Bhavya Kailkhura, Zhiding Yu, and Z Morley Mao. Modelnet40-c: Arobustness benchmark for 3d point cloud recognition under corruption.

[36] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 403–412, 2019.

[37] 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.

[38] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *International Conference on Computer Vision (ICCV)*, 2019.

[39] 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.

[40] Xin Wei, Ruixuan Yu, and Jian Sun. View-gcn: View-based graph convolutional network for 3d shape analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1850–1859, 2020.

[41] 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.

[42] 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.

[43] Ming Zhu, Chao Ma, Pan Ji, and Xiaokang Yang. Cross-modality 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3772–3781, 2021.