# What Makes for Effective Few-shot Point Cloud Classification?

Chuangguan Ye[1], Hongyuan Zhu[2*], Yongbin Liao[1], Yanggang Zhang[1], Tao Chen[1†], Jiayuan Fan[3]
[1]School of Information Science and Technology, Fudan University, China
[2]Institute for Infocomm Research, A*STAR, Singapore
[3]Academy for Engineering and Technology, Fudan University, China
{cgye19,19210720121,ygzhang19,eetchen,jyfan}@fudan.edu.cn, hongyuanzhu.cn@gmail.com

## Abstract

*Due to the emergence of powerful computing resources and large-scale annotated datasets, deep learning has seen wide applications in our daily life. However, most current methods require extensive data collection and retraining when dealing with novel classes never seen before. On the other hand, we humans can quickly recognize new classes by looking at a few samples, which motivates the recent popularity of few-shot learning (FSL) in machine learning communities. Most current FSL approaches work on 2D image domain, however, its implication in 3D perception is relatively under-explored. Not only needs to recognize the unseen examples as in 2D domain, 3D few-shot learning is more challenging with unordered structures, high intra-class variances and subtle inter-class differences. Moreover, different architectures and learning algorithms make it difficult to study the effectiveness of existing 2D methods when migrating to the 3D domain.*

*In this work, for the first time, we perform systematic and extensive studies of recent 2D FSL and 3D backbone networks for benchmarking few-shot point cloud classification, and we suggest a strong baseline and learning architectures for 3D FSL. Then, we propose a novel plug-and-play component called Cross-Instance Adaptation (CIA) module, to address the high intra-class variances and subtle inter-class differences issues, which can be easily inserted into current baselines with significant performance improvement. Extensive experiments on two newly introduced benchmark datasets, ModelNet40-FS and ShapeNet70-FS, demonstrate the superiority of our proposed network for 3D FSL.*

## 1. Introduction

3D point cloud object classification is an important task for various computer vision, and is widely applied in many scenarios, like robotics [34], indoor simultaneous localization and mapping (SLAM) [54], autonomous vehicles (AVs) [38, 16], etc. unlike traditional point cloud recognition algorithms that extract handcrafted features [12, 53, 33, 44], deep-learning based methods can learn more representative features from shape projections [51, 24, 7] or raw points [23, 52, 18, 17] with deep networks, achieving better performance in kinds of point cloud processing tasks.

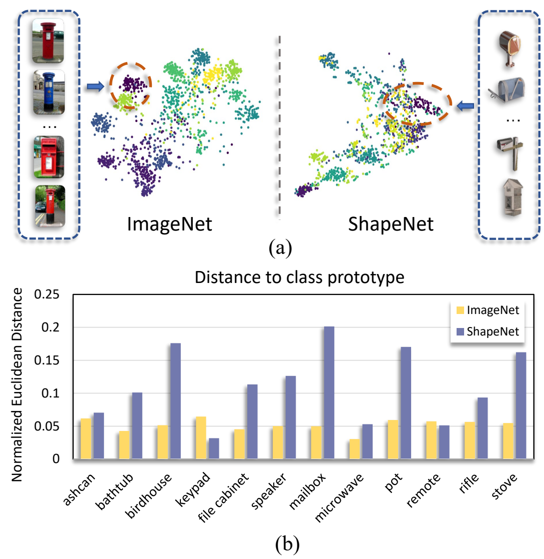


Figure 1. The challenge of few-shot 3D point cloud classification. (a) is the T-SNE on ImageNet [31] and ShapeNet [3] using pre-trained 2D image and 3D point cloud feature respectively with the **Mailbox** class highlighted. (b) The mean normalized Euclidean distance between each example and its class prototype in the visual and point cloud embedding space respectively. The embedding quality of 2D visual domain is much higher than 3D point cloud domains because image-based pre-trained models [5], such as ResNet, use deeper networks trained on millions of images, whereas point cloud-based models, such as PointNet, use shallower networks trained on only a few hundred point clouds with high intra-class variations and subtle inter-class differences.

However, there are two crucial problems of deep learning based approaches for point cloud classification. Firstly, deep networks are usually data-driven and strongly rely on

[*]This work was completed under the supervision of Dr. Hongyuan Zhu at A*STAR.
[†]Tao Chen is the corresponding author.

large amounts of labeled training data, while the process of annotating data is cumbersome and costly. Secondly, the networks tend to have poor generalization ability for novel classes never seen before. Data-augmentation based methods [35, 6, 27] and regularization techniques [11, 50] can alleviate the data deficiency problem without adding new labeled data. But these methods may not get promising results on unseen classes or new tasks without sufficient labeled training data. In recent years, few-shot learning (FSL) algorithms [45, 39] are introduced to rapidly generalize deep networks to new tasks with only one or few annotated examples, having great success in the 2D image domain. By constructing many auxiliary training meta-tasks, these 2D FSL algorithms use metric learning [39, 45] or optimization-based approaches [13] to learn transferable knowledge and propagate them to new tasks.

Compared to the FSL in the 2D image domain, its study on 3D data is still relatively under-explored with the following challenges: 1) point cloud is a set of irregular, unstructured and unordered points defined in European space [2], to what extent existing point cloud architecture and learning algorithm can perform for 3D FSL is unknown; 2) most 2D FSL algorithms can learn more discriminative representations using deeper networks pre-trained on large-scale base class data [4, 5], e.g. tieredImageNet [29] contains 608 classes and has a total number of 800,000 samples. By contrast, most 3D model datasets, e.g. ModelNet [48] and ShapeNet [3], have a much smaller number of labeled data (ModelNet [48] includes 40 classes with a total number of 12,311 samples). So the point-based models trained on low volume of data may generate poor-quality 3D feature clusters with high intra-class variations and subtle inter-class differences, as shown in Figure 1. Therefore, how to address these issues requires further exploration.

In this paper, we study 3D few-shot learning (3D FSL) in a systematic manner for the first time. We perform extensive studies and discussions of various state-of-the-art point cloud architectures and few-shot learning algorithms in the context of 3D FSL, and propose a strong baseline for 3D FSL. Furthermore, to address the issues of high intra-class variance and subtle inter-class differences, we propose a plug-and-play Cross-Instance Adaption (CIA) module for 3D FSL. The CIA contains two modules called Self-Channel Interaction (SCI) module and Cross-Instance Fusion (CIF) module, which can be flexibly inserted into most current FSL algorithms with few changes and achieves significant performance improvement. Moreover, for objectively evaluating, we introduce new splits of ModelNet [48] and ShapeNetCore [3] respectively, and construct two benchmark datasets, ModelNet40-FS, ShapeNet70-FS for 3D few-shot point cloud classification. Codes and datasets will be released for facilitating future research in this area.

Our contributions can be summarized as:

- We are the first to perform a systematic study of 3D few-shot learning (FSL) in terms of the network architectures and propose a strong baseline for 3D FSL.
- We propose a novel plug-and-play Cross Instance Adaption (CIA) module that can be flexibly inserted into most current FSL algorithms/backbones and achieves significant improvement for 3D point cloud few-shot object classification.
- The proposed method achieves the state-of-the-art performance on the two newly built 3D FSL benchmark datasets ModelNet40-FS and ShapeNet70-FS under different settings.

## 2. Problem Definition and Related Work

Let $(x, y)$ denotes a point cloud instance $x$ and its label $y$, where $x = \{p_j | j = 1, ..., n\}$ is an unordered point set including $n$ points, and each point $p_j$ is represented by its 3D coordinates. In standard $N$-way-$K$-shot FSL setting [4], the goal of FSL is to train a predictor for query examples with few labeled support examples, where the labeled support examples are denoted as support set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{N_s = N\times K}$ containing $N$ classes with $K$ examples for each class, and the query examples are denoted as query set $\mathcal{Q} = \{(x_i, y_i)\}_{i=1}^{N_q = N\times Q}$ containing the same $N$ classes with $Q$ examples for each class.

For few-shot point cloud classification, meta-learning is the state-of-the-art paradigm with a set of meta-training episodes $\mathcal{T} = \{(\mathcal{S}_i, \mathcal{Q}_i)\}_{i=1}^{I}$ by optimizing following objectives:

$$\theta^*, \phi^* = \underset{\theta,\phi}{\arg\min} \mathcal{L}\left(\mathcal{T}; \theta, \phi\right). \tag{1}$$

where $\mathcal{T}$ are sampled from the training set and $\mathcal{L}$ denotes the cross-entropy loss function defined as:

$$\mathcal{L}\left(\mathcal{T}; \theta, \phi\right) = \mathbb{E}_{\mathcal{T}}\left[-\log p\left(\hat{y} = c|x\right)\right], \tag{2}$$

with the prediction $p\left(\hat{y} = c|x\right)$ can be given by:

$$p\left(\hat{y} = c|x\right) = softmax\left(\mathcal{C}_\theta\left(\mathcal{F}_\phi\left(x\right)\right)\right), \tag{3}$$

where $x$ is the input point cloud instance, $\hat{y}$ is the predicted label. The $\mathcal{F}$ is the embedding network parameterized by $\phi$ and $\mathcal{C}$ is the classifier parameterized by $\theta$.

Once meta-training is finished, generalization of the predictor is evaluated on meta-testing episodes $\mathcal{V} = \{(\mathcal{S}_j, \mathcal{Q}_j)\}_{j=1}^{J}$, sampled from the testing set. Note that we denote the classes in $\mathcal{V}$ as novel classes, which are disjoint with the base classes in meta-training episodes $\mathcal{T}$.

Therefore, there are two key challenges in Point Cloud FSL: 1) how to represent the point cloud data properly for few-shot learning; 2) how to effectively transfer the knowledge gained in meta-training episodes to meta-testing episodes with a small number of labeled samples with high intra-class variance and subtle inter-class differences. In the following section, we'll introduce the recent efforts in 3D point cloud classification and 2D few-shot learning.

## 2.1. Related Work

**3D point cloud classification:** Unlike conventional handcrafted feature extraction approaches [44, 12], deep-learning based methods can learn more complicated and representative features with deep networks. According to the way of feature extraction from structured grid or raw points, existing methods can be generally divided into projection-based and point-based. Projection-based networks first convert the irregular points into a structured representation, such as multi-view projection [42, 51], voxel [20, 30] or lattices [41, 26], and then apply the typical 2D or 3D convolution neural networks (CNN) to extract view-wise or structural features. However, these methods may suffer from explicit information loss or higher memory consumption [10].

In contrast, directly applying deep networks on raw points is becoming a trend in recent years. PointNet [23] is the first deep network for unstructured point cloud, which learns point-wise features with multilayer perceptron (MLP) and aggregates global feature with a symmetric function implemented by a max-pooling layer. After that, lots of works such as PointNet++ [25], PointCNN [15], RSCNN [18], DensePoint [17], and DGCNN [47] explore the local relations in a specific region with convolution-based networks or graph-based networks, which can capture meaningful geometric features and achieve the state-of-the-art performance. Nonetheless, these deep learning approaches are greedy for the availability of massive annotated data, and may have poor generalization on novel classes that are unseen during training.

Recently, there are some works that also consider 3D point cloud learning with a small number of training data. Sharma *et al.* [37] mainly studies feature representation learning using self-supervision which is orthogonal to our few-shot learning. LSSB [40] try to learn a discriminative embedding space for 3D model multi-view images with the bias of point cloud shape.

Different from these works, we are the first to study few-shot point cloud classification in a systematic manner by reviewing recent 3D point cloud learning networks, and suggest strong baselines for the problem. We also propose an effective plug-and-play module, Cross Instance Adaption module, to address the high intra-class variance and subtle inter-class differences issues of 3D FSL, which can be flexibly inserted into most current FSL algorithms.

**2D few-shot learning:** With the characteristics of less annotated training data and good generalization on new tasks, few-shot learning is a promising direction for deep learning. In general, existing FSL algorithms is based on meta-learning framework and can be roughly categorized into metric-based methods [45, 39, 43, 14, 22] and optimization-based methods [8, 13, 28, 21, 32].

Metric-based methods focus on learning an embedding space where similar samples pairs are closer, or designing a metric function to compare the feature similarity of samples. Matching Network [45]adapts a bidirectional LSTM module to get full context embeddings and uses cosine distance to classify query samples. Prototypical Network [39] on the other hand first averages the support-set features for each class as a class prototype and then takes squared Euclidean distance to measure the similarity with query samples, which demonstrates better performance than MatchingNet. Relation Net [43] further proposes a learnable metric module to get relation scores between support set and query set.

Optimization-based methods, on the other hand, regard meta-learning as an optimization process. MAML [8] learns a model-agnostic initialization parameter that can produce great improvement with few gradient steps on new tasks. MetaOptNet [13] incorporates a differentiable quadratic programming solver to learn a feature-relevant linear SVM predictor which can offer better generalization for novel categories.

In this work, we perform a systematic study of different meta learning algorithms under few-shot point cloud classification tasks, suggest strong baselines and components for the problem.

# 3. Empirical Study

In the following sections, we would like to perform empirical study of recent state-of-the-art few-shot learning methods on point cloud data with different popular point-based network backbones on our newly proposed ModelNet40-FS and ShapeNet70-FS datasets. The experimental settings and implementation details can be found in Section 5.

## 3.1. State-of-the-art 2D FSL on Point Cloud

We first analyze recent state-of-the-art 2D few shot learning methods' performance, when they migrate to few-shot point cloud classification task on our newly proposed benchmark datasets. Given its simplicity and efficiencies, we adopt PointNet [23] as backbone for feature embeddings. Specifically, we divide the state-of-the-arts into following groups:

- Metric-based methods **M**: ProtoNet [39], Relation Net [43], FSLGNN [36],
- Optimization-based methods **O**: Meta-learner [28], MAML [8], MetaOptNet [13]

The comparison results of the metric-based methods and optimiza-tion-based methods are shown in Table 1. One

| Method | | ModelNet40-FS | | ShapeNet70-FS | | 5way-1shot-15query | | |
|---|---|---|---|---|---|---|---|---|
| | | 5w-1s | 5w-5s | 5w-1s | 5w-5s | PN | GFLOPs | TPS |
| M | ProtoNet [39] | **65.31 $\pm$ 0.78** | 79.04 $\pm$ 0.54 | **65.96 $\pm$ 0.81** | **78.77 $\pm$ 0.67** | **0.15M** | **6.16** | **118.23** |
| | Relation Net [43] | 64.10 $\pm$ 0.72 | 75.75 $\pm$ 0.57 | 65.88 $\pm$ 0.85 | 76.25 $\pm$ 0.71 | 0.28M | 6.50 | 96.03 |
| | FSLGNN [36] | 59.69 $\pm$ 0.73 | 76.06 $\pm$ 0.63 | 64.98 $\pm$ 0.84 | 76.14 $\pm$ 0.73 | 2.23M | 10.44 | 69.02 |
| O | Meta-learner [28] | 58.69 $\pm$ 0.81 | 76.60 $\pm$ 0.65 | 62.64 $\pm$ 0.91 | 73.10 $\pm$ 0.80 | 0.88M | 6.28 | 9.14 |
| | MAML [8] | 57.58 $\pm$ 0.89 | 77.95 $\pm$ 0.62 | 59.20 $\pm$ 0.88 | 75.10 $\pm$ 0.75 | 0.68M | 6.19 | 38.71 |
| | MetaOptNet [13] | 64.99 $\pm$ 0.87 | **79.54 $\pm$ 0.61** | 65.08 $\pm$ 0.89 | 77.81 $\pm$ 0.75 | **0.15M** | **6.16** | 11.95 |

Table 1. Few-shot point cloud classification results with 95% confidence intervals on ModelNet40-FS and ShapeNet70-FS with PointNet [23] as backbone. Bold denotes the best result and underline represents the second best. $M$ is the metric based methods and $O$ is the optimization based methods. PN: Parameter Number. GFLOPs: the number of floating-point operations. TPS: inference Tasks per Second on NVIDIA 2080Ti GPU, with 15 query examples for each task.

| Method | Backbone | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|---|
| | | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| ProtoNet [39] | PointNet [23] | 65.31 | 79.04 | 65.96 | 78.77 |
| | PointNet++ [25] | 64.96 | 83.66 | 66.33 | 80.95 |
| | PointCNN [15] | 60.38 | 76.95 | 64.02 | 76.34 |
| | RSCNN [18] | 69.72 | 84.79 | 68.66 | **82.55** |
| | DensePoint [17] | 66.99 | 82.85 | 65.81 | 80.74 |
| | DGCNN [47] | **69.95** | **85.51** | **69.03** | 82.08 |

Table 2. Results of ProtoNet [39] on ModelNet40-FS and ShapeNet70-FS with different 3D backbone architectures. Bold denotes the best result and underline represents the second best.

can observe that ProtoNet [39] with PointNet [23] backbone can achieve top performance at 65.31% and 65.96% respectively at both datasets and still has large room for further improvement.

Moreover, one can conclude that metric-based methods outperform the optimization-based methods in point cloud scenario. One possible explanation is that the optimization-based methods are more sensitive to neural network architecture and require arduous hyperparameter setting to achieve good generalization, as evidenced by recent research in 2D image domains [1].

It's worth noting that ProtoNet [39] and MetaOptNet [13] have the same number of parameters, because they employ parameter-free square Euclidean distance and SVM as meta classifier respectively. However, solving the quadratic programming of SVM in MetaOptNet [13] is very computationally expensive. In brief, ProtoNet [39] has a better trade-off between classification accuracy and algorithm complexity with a high inference speed.

### 3.2. Influence of Backbone Architecture on FSL

Then we further study the influence of the backbones to point cloud FSL in ProtoNet [39]. We select three types of current state-of-the-art 3D point-based networks including:

- Pointwise-based: PointNet [23] and PointNet++ [25].
- Convolution-based: PointCNN [15], RSCNN [18] and DensePoint [17].
- Graph-based: DGCNN [47].

For fair comparisons, we remove the last fully connected layers and train the networks from scratch with the same training strategy as suggested by these methods. After that, we feed the embedding features generated by these network into ProtoNet [39] for few-shot classification and the comparison results are shown in Table 2. One can conclude that the graph-based network DGCNN [47] achieves higher classification accuracy than other networks on these two datasets. The reason may be that graph-based methods dynamically updates the point-wise connection graph in feature space and extracts edge features with EdgeConv layers, hence can learn more discriminative features.

Therefore, we will use ProtoNet [39] with DGCNN [47] as the strong baseline for benchmarking with 3D FSL.

## 4. Approach

### 4.1. Overview

Given the baseline in the aforementioned section, 3D FSL still faces certain challenges: 1) there are strong intra-class variations and subtle inter-class differences among support set and query set with a small amount of data; 2) many FSL methods extract features from support and query examples independently without considering the correlations between these two sets, therefore current features learning are not discriminative enough.

To address these challenges, we propose a novel plug-and-play Cross Instance Adaption (CIA) module, which can be inserted into existing backbones and learning frameworks to learn more discriminative representations for the support set and query set, to be elaborated in Section 4.2. By integrating CIA module with current meta-learning framework, we come up a novel and strong network for 3D FSL classification, which is illustrated in Figure 2.

First, the embedding module $\mathcal{F}_\phi$ takes support set $\mathcal{S}$ and query set $\mathcal{Q}$ as input, and maps each point cloud example $x \in \mathbb{R}^{n\times 3}$ as a feature vector $f = \mathcal{F}_\phi\left(x\right)$, where $f \in \mathbb{R}^{1\times d}$. Then we define the prototype feature for class $c_i$ in support set as $f_p^i = \frac{1}{|K|}\Sigma_{x_s\in c_i}\mathcal{F}_\phi\left(x_s\right)$ as the mean of its $K$ support examples, and the query feature for a query example $x_q^j$ as $f_q^j = \mathcal{F}_\phi\left(x_q^j\right)$, where $i \in [1, N]$ and $j \in [1, N_q]$.

However, the first step extracts features from support set and query set separately. It also ignores the high intra-class variance and subtle inter-class differences issues in 3D FSL. Hence, the learned support and query features are not discriminative enough with huge distribution shift, as demonstrated in Figure 3 (a). Therefore, we propose to update the prototype features $f_p^i$ and query feature $f_q^j$ by feeding them into the novel Cross Instance Adaption (CIA) mod-

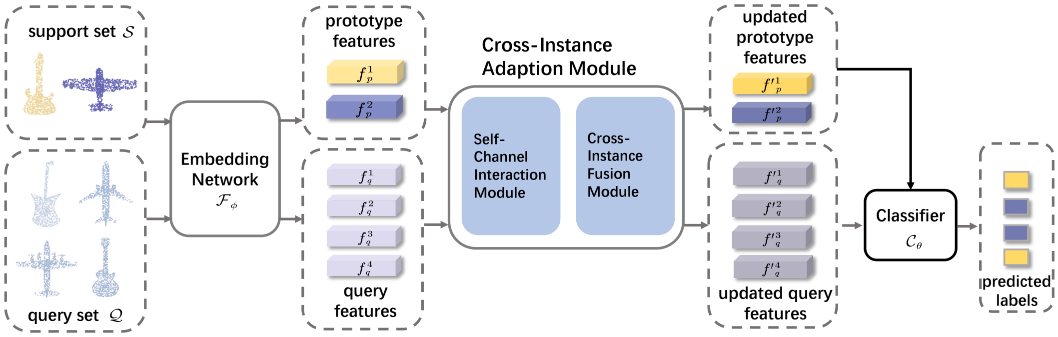


Figure 2. An illustration of our proposed framework for 3D few-shot point cloud classification with an introduction in Section 4.1. For clarity we only present the 2-way 1-shot 2-query setting.

ule in Section 4.2, to learn more diverse and discriminative support and query features $f'^{\,i}_p$ and $f'^{\,j}_q$, and to mitigate the distribution shift for better classification, as demonstrated in Figure 3 (d).

After that, we take Square Euclidean Distance metric function as classifier $\mathcal{C}_\theta$ to measure the distance between each class prototype and query examples in the feature space. The probability of predicted label $\hat{y}_j$ for $f'^{\,j}_q$ as class $c_i$ is denoted as:

$$p\left(\hat{y}_j = c_i | f'^{\,j}_q\right) = \frac{\exp\left(-d\left(f'^{\,j}_q, f'^{\,c_i}_p\right)\right)}{\sum_{i=1}^{N}\exp\left(-d\left(f'^{\,j}_q, f'^{\,i}_p\right)\right)}, \quad (4)$$

where $d\left(.,.\right)$ is the Square Euclidean Distance, $f'^{\,i}_p$ and $f'^{\,j}_q$ are the updated features generated by CIA module. Finally, we can get the cross-entropy loss with Equation 2, and optimize the network end-to-end by minimizing the Equation 5:

$$L_{CE} = -\frac{1}{N}\frac{1}{N_q}\sum_{i}^{N}\sum_{j}^{N_q}\mathbb{1}\left[y_j = c_i\right]\log\left(p\left(\hat{y}_j = c_i|f'^{\,j}_q\right)\right), \quad (5)$$

where, $N$ and $N_q$ are the number of class prototypes and query examples respectively, $y_j$ is the ground truth of $f'^{\,j}_q$, and $\mathbb{1}$ denotes the Kronecker delta function.

### 4.2. Cross-Instance Adaption Module

To learn more discriminative features, Cross Instance Adaption module (CIA) consists of two modules: first, an Self-Channel Interaction (SCI) module is designed to learn diverse and discriminative features of a point clouds object by modeling channel correlations to address the issues of subtle inter-class differences, and then a Cross-Instance Fusion (CIF) module is designed to explore instance-wise interaction to address high intra-class variances issues, which can compensate the prototypical information and rectify the feature distribution by re-weighting support and query features with a meta-learner.

#### 4.2.1 Self-Channel Interaction Module

3D FSL faces the challenge of subtle inter-class differences, for example 'chair' and 'bench' can both have samples with

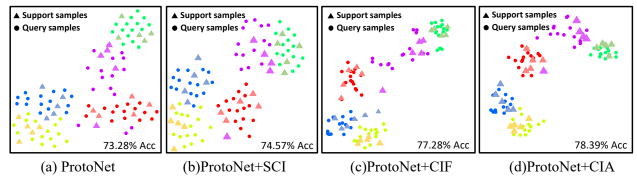


Figure 3. The t-SNE comparison of feature distribution of support set and query set before and after using the CIA modules (SCI and CIF) in Section 4.2. △ stands for the support features, ● represents the query features.

handles and cushion. [49, 9] point out that different channels can convey different semantic information. Inspired by this observation, we attempt to model channel-wise relationships to learn more diverse features to address subtle inter-class differences with an illustration in Figure 4.

Concretely, a query-vector $q \in \mathbb{R}^{1\times d}$ and a key-vector $k \in \mathbb{R}^{1\times d}$ are firstly generated from the embedding feature vector $f$ with two linear embedding functions parameterized by $\varphi$ and $\gamma$ respectively. Then Channel Interaction Module is executed by using a bilinear operation between $q^T$ and $k$ to get a channel-wise relation score map:

$$R = q^T k,\ R \in \mathbb{R}^{d\times d}. \quad (6)$$

Then we normalize each column of $R$ by performing a softmax function to get the weight matrix $R'$. Specially, the $i^{th}$ value of the $j^{th}$ column in $R'$ can be calculated by:

$$R'_{ij} = \frac{\exp\left(-R_{ij}\right)}{\sum\limits_{k=1}^{d}\exp\left(-R_{kj}\right)},\ R' \in \mathbb{R}^{d\times d}, \quad (7)$$

where $\sum\limits_{k=1}^{d} R'_{kj} = 1$. After that, we use the channel-wise relation score map $R'$ to re-weight the initial feature $f$ and get vector $v$, which can be denoted by:

$$v = fR',\ v \in \mathbb{R}^{1\times d}. \quad (8)$$

Note that the $i^{th}$ channel of $v$, defined as $v_i = f_1 R'_{i1} + ... + f_d R'_{id}$, is the weighted sum of all channels in $f$, so higher value of $v_i$ indicates that the $i^{th}$ channel is more informative. Finally, we further combine $v$ and $f$ to compensate the discarded information and output the updated

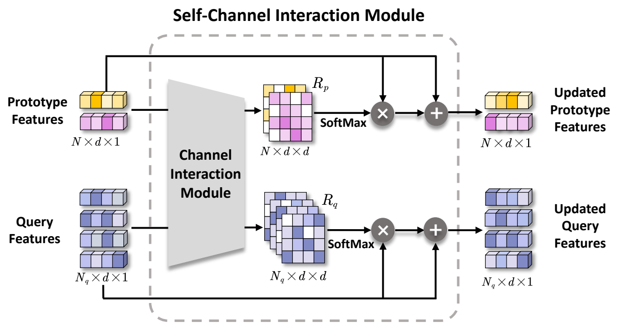


Figure 4. An illustration of the Self-Channel Interaction Module. For clarity we only present the 2-way 1-shot 2-query setting.

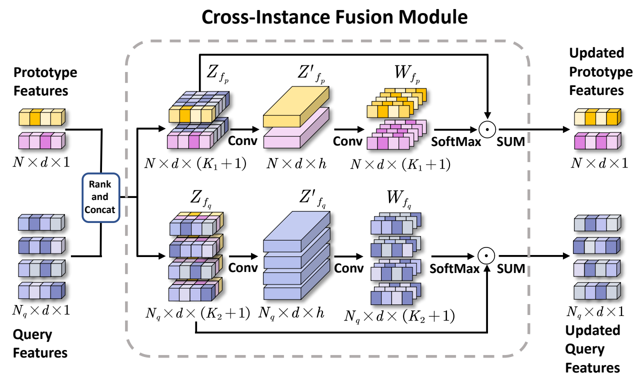


Figure 5. An illustration of the Cross-Instance Fusion Module. For clarity we only present the 2-way 1-shot 2-query setting. The $\odot$ is the element-wise product.

features $f'$:

$$f' = v + f,\ f' \in \mathbb{R}^{1\times d} \tag{9}$$

After using the SCI module, the features become more discriminative to separate different classes, as shown in Figure 3 (b). More visualization analysis of SCI module can be found in Supplementary Material.

#### 4.2.2 Cross-Instance Fusion Module

Most existing methods [45, 39, 43, 22] extract support features and query features independently, as a result there are huge feature distribution shift (as shown in Figure 3 (a)) exists between support set and query set due to the low volume of data and high intra-class variance issues. Thus, we propose a simple but effective Cross-Instance Fusion (CIF) module with a meta-network to re-weight and update support features and query features by considering their instance-wise relationships.

Specifically, as illustrated in Figure 5, we first concatenate each prototype feature $f_p^i$ (where $i \in [1, N]$) to its top $K_1$ cosine similar query features, and get $Z_{f_p^i}$:

$$Z_{f_p^i} = \left[f_p^i, f_q^{\langle top1\rangle}, ..., f_q^{\langle topK_1\rangle}\right], Z_{f_p^i} \in \mathbb{R}^{1\times d\times (K_1+1)}, \tag{10}$$

where $[\cdot]$ is the concatenation operation, $d$ is the number of feature channels, $K_1 \leq N_q$, and $f_q^{\langle top1\rangle}$ represents the query feature having the highest cosine similarity with prototype feature $f_p^i$. Similarly, we concatenate each query feature $f_q^j$ (where $j \in [1, N_q]$) to its top $K_2 (\leq N)$ cosine similar prototype features, and get $Z_{f_q^j}$:

$$Z_{f_q^j} = \left[f_q^j, f_p^{\langle top1\rangle}, ..., f_p^{\langle topK_2\rangle}\right], Z_{f_q^j} \in \mathbb{R}^{1\times d\times (K_2+1)}. \tag{11}$$

We then employ two $1\times 1$ convolution layers as a metaleaner to encode the concatenated features and generate a weight matrix $W_f$ for $Z_f$. After that, we update the prototype (query) features by using the weighted sum of $Z_f$ instead of simple averaging, which can fuse the instance-wise information flexibly. For example, the weight matrix $W_{f_p^i}$ for $Z_{f_p^i}$ is denoted as:

$$W_{f_p^i} = f_2\left(f_1\left(Z_{f_p^i}\right)\right), W_{f_p^i} \in \mathbb{R}^{1\times d\times (K_1+1)}. \tag{12}$$

And the prototype feature $f_p^i$ can be updated by combining the $K_1$ concatenated features in $Z_{f_p^i}$ based on $W_{f_p^i}$:

$$f_p'^i = \sum\left(softmax\left(W_{f_p^i}\right) \odot Z_{f_p^i}\right), f_p'^i \in \mathbb{R}^{1\times d} \tag{13}$$

where $f_1(\cdot)$ is the first $1\times 1$ Conv layer encoding the $Z_{f_p^i}$ into a $h$-dim feature interaction $Z'_{f_p^i}$, and the second $1\times 1$ Conv layer $f_2(\cdot)$ is designed to adjust the dimension of interaction $Z'_{f_p^i}$ to generate the weight matrix $W_{f_p^i}$, and $\odot$ is the element-wise product.

Similarly, we can easily get $W_{f_q^j} = f_4\left(f_3\left(Z_{f_q^j}\right)\right) \in \mathbb{R}^{1\times d\times (K_2+1)}$ for $Z_{f_q^j}$ with two $1\times 1$ Conv layers $f_3(\cdot)$ and $f_4(\cdot)$, whose parameters are not shared with $f_1(\cdot)$ and $f_2(\cdot)$, and generate the updated query features $f_q'^j$.

After using the CIF module, the distribution shift between support and query set is mitigated, as shown in Figure 3 (c). More visualization analysis of CIF module can be found in Supplementary Material.

## 5. Experiments

In this section, first, we introduce two datasets for few-shot 3D point cloud classification and describe the implementation details. Then, we compare the state-of-the-art FSL algorithms with DGCNN [47] as backbone using CIA modules on these datasets. Finally, we conduct extensive ablation studies to evaluate the effectiveness of the CIA module when inserted into different algorithms.

### 5.1. Dataset

ModelNet40 [48] and ShapeNetCore [3] are two standard benchmark datasets for 3D point cloud learning. However, exiting splits of training set and testing set are class-overlapping, which should be disjunctive in the few-shot

| Backbone | Method | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|---|
| | | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| DGCNN [47] | ProtoNet [39] | 69.95 ± 0.67 | 85.51 ± 0.52 | 69.03 ± 0.84 | 82.08 ± 0.72 |
| | Relation Net [43] | 68.57 ± 0.73 | 82.01 ± 0.53 | 67.87 ± 0.86 | 77.99 ± 0.70 |
| | FSLGNN [36] | 61.96 ± 0.76 | 80.22 ± 0.55 | 66.25 ± 0.88 | 76.20 ± 0.77 |
| | Meta-learner [28] | 59.08 ± 0.86 | 76.99 ± 0.67 | 64.53 ± 0.83 | 74.61 ± 0.72 |
| | MAML [8] | 62.57 ± 0.88 | 77.41± 0.73 | 64.39 ± 0.76 | 74.11 ± 0.68 |
| | MetaOptNet [13] | 67.05 ± 0.78 | 85.05 ± 0.59 | 68.27 ± 0.93 | 81.06 ± 0.76 |
| | **Ours** | **75.70 ± 0.74** | **87.15 ± 0.47** | **73.57 ± 0.81** | **83.24 ± 0.67** |

Table 3. Few-shot point cloud classification results with 95% confidence intervals on ModelNet40-FS and ShapeNet70-FS with DGCNN [47] as backbone. Bold denotes the best result and underline represents the second best.

setting [4, 45]. To meet the requirement, we propose new splits of ModelNet40 and ShapeNetCore, and build two benchmarks, ModelNet40-FS and ShapeNet70-FS, for 3D few-shot point cloud classification. We carefully split the dataset according to the number of instances in each category, making sure that data distributions of the training set and testing set are similar. Concretely, **ModeNet40-FS** includes 30 training classes with 9,240 examples and 10 testing classes with 3,104 examples. **ShapeNet70-FS** has a larger number of data, which totally contains 30,073 examples, 50 classes with 21,722 samples for training and 20 classes with 8,351 samples for testing. Details of these splits are listed in Supplementary Material.

### 5.2. Training and Testing Details

We follow the standard episode-based FSL setting [4] to train and evaluate. We first train the network from scratch with 80 epochs, and each epoch contains 400 meta-training episodes and 600 validating episodes randomly sampled from the training set. Each episode consists of $N$ classes with $K$ labeled support examples and $Q$ query examples for each class, which is denoted as the $N$-way $K$-shot $Q$-query setting. Once the meta-training is ended, we test the network with 700 meta-testing episodes which are randomly sampled from the testing set with the same $N$-way $K$-shot $Q$-query setting. We average the classification results of these meta-testing episodes with 95% confidence intervals as the final performance. Specially, we employ 5-fold cross-validation, where the training set will be randomly divided into 5 even subsets, and each subset is used once as a validation set. Finally, we report the average performance of the 5 estimations on the testing set.

### 5.3. Implementation Details

We take DGCNN [47] as the embedding network, consisting of four EdgeConv layers (64,64,128,256) and an MLP layer. More adapted details of backbone networks and FSL algorithms can refer to Supplementary Material. Each point cloud instance consists of 512 points sampled randomly from the CAD model surface. We use Adam optimizer with an initial learning rate of 0.0008 and gamma of 0.5. The learning rate declines every 5 epochs. We also apply random points jittering and rotating to augment data as in [23] during training. Other specific implementation details are described in the following subsections.

| Method | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|
| | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| ProtoNet [39] | 69.95 | 85.51 | 69.03 | 82.08 |
| ProtoNet [39]+CIA | **75.70** | **87.15** | **73.57** | **83.24** |
| Relation Net [43] | 68.57 | 82.01 | 67.87 | 77.99 |
| Relation Net [43]+CIA | **70.55** | **83.59** | **68.67** | **78.60** |
| FSLGNN [36] | 61.96 | 80.22 | 66.25 | 76.20 |
| FSLGNN [43]+CIA | **63.81** | **83.57** | **67.40** | **78.62** |
| Meta-learner [28] | 59.08 | 76.99 | 64.53 | 74.61 |
| Meta-learner [28]+CIA | **60.55** | **77.30** | **65.61** | **75.01** |
| MAML [8] | 62.57 | 77.41 | 64.39 | 74.11 |
| MAML [8]+CIA | **63.32** | **78.29** | **65.25** | **75.03** |
| MetaOptNet [13] | 67.05 | 85.05 | 68.27 | 81.06 |
| MetaOptNet [13]+CIA | **74.70** | **87.10** | **72.82** | **83.08** |

Table 4. Comparisons of the classification results after incorporating CIA Module into different FSL algorithms on ModelNet40-FS and ShapeNet70-FS with DGCNN [47] as backbone.

### 5.4. Comparison with the Baselines

To verify the effectiveness of our proposed network, we first compare the classification accuracy with the aforementioned FSL baselines on two benchmark datasets, ModelNet40-FS and ShapeNet70-FS. The results in Table 3 show that our method exceeds other baselines by a large margin, about 5% for 1-shot and 1.5% for 5-shot and outperforms other baselines on both two datasets.

One potential explanation is that ProtoNet is a metric-based approach that predicts labels for query examples based on the nearest square Euclidean distance with each class prototype. However, features for support and query sets are extracted independently. The CIA Module can adjust the distribution of support and query examples in the feature space by considering feature-level and instance-level association, which could enhance the discrimination between prototypes and query examples.

### 5.5. Ablative Analysis

In this section, we conduct ablation studies to analyze the effects of various designs of the proposed module for few-shot point cloud classification.

**Adding CIA Module to Different FSL Baselines.** We embed the CIA Module into metric-based and optimization-based FSL baselines to validate the generalization ability with multiple systems. All the experiments take DGCNN as backbone for features extraction and share the same training strategy for a fair comparison. Table 4 shows the improvement of adapting the CIA Module into each algorithm. One can observe that there is an approximately 2% consistency increase after incorporating with CIA Module, demonstrat-

| Backbone | Method | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|---|
| | | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| PointNet [23] | PN | 65.31 | 79.04 | 65.96 | 78.77 |
| | PN+CIA | **67.40** | **80.31** | **69.36** | **80.31** |
| PointNet++[25] | PN | 64.96 | 83.66 | 66.33 | 80.95 |
| | PN+CIA | **72.42** | **84.93** | **72.66** | **82.65** |
| PointCNN [15] | PN | 60.38 | 76.95 | 64.02 | 76.34 |
| | PN+CIA | **64.20** | **80.86** | **66.12** | **78.06** |
| RSCNN [18] | PN | 69.72 | 84.79 | 68.66 | 82.55 |
| | PN+CIA | **74.23** | **85.58** | **73.86** | **83.85** |
| DensePoint [17] | PN | 66.99 | 82.85 | 65.81 | 80.74 |
| | PN+CIA | **72.36** | **84.41** | **72.48** | **81.84** |
| DGCNN [47] | PN | 69.95 | 85.51 | 69.03 | 82.08 |
| | PN+CIA | **75.70** | **87.15** | **73.57** | **83.24** |

Table 5. Comparisons of the classification results after incorporating CIA Module into ProtoNet(PN) [39] with different backbones on ModelNet40-FS and ShapeNet70-FS.

| ProtoNet | SCI | CIF | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|---|---|
| | | | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| ✓ | | | 69.95 | 85.51 | 69.03 | 82.08 |
| ✓ | ✓ | | 70.51 | 85.96 | 69.37 | 82.32 |
| ✓ | | ✓ | 73.67 | 86.69 | 72.32 | 82.73 |
| ✓ | ✓ | ✓ | **75.70** | **87.15** | **73.57** | **83.24** |

Table 6. Ablation study of CIA Module with DGCNN [47] as embedding network. "SCI" denotes the Self-Channel Interaction block and "CIF" means the Cross-Instance Fusion block.

| Method | ModelNet40-FS | | ShapeNet70-FS | |
|---|---|---|---|---|
| | 5w-1s | 5w-5s | 5w-1s | 5w-5s |
| SimpleShot [46] | 61.87 | 78.17 | 61.58 | 74.63 |
| SimpleShot /w SB [40] | 63.33 | 76.41 | 64.45 | 73.77 |
| SSL3DFSL [36] | 64.89 | 79.59 | 65.76 | 79.19 |
| **Ours** | **75.70** | **87.15** | **73.57** | **83.24** |

Table 7. The comparisons of other 3DFSL methods on ModelNet40-FS and ShapeNet70-FS with DGCNN [47] as backbone.

ing that the proposed module can improve both metric-based and optimization-based FSL baselines significantly.

**Adding CIA Module into Different Backbones.** We also verify the effectiveness of the CIA Module adapting into ProtoNet with different 3D support backbones aforementioned in Table 5. We first remove the FC layers of these 3D networks and feed the output feature vectors into the CIA Module to generate re-weighted features. Then we employ square Euclidean distance as the metric function to classify the unlabeled query samples. The comparison results in Table 5 show that after including CIA Module, different state-of-the-art backbones can achieve significant performance improvement.

**Ablation Study of CIA modules.** After that, we perform ablation studies to quantify the contribution of two proposed modules in CIA with the results listed in Table 6. We can clearly observe that both "SCI" and "CIF" could provide positive impacts and improve the performance for ProtoNet on MdoelNet40-FS and ShapeNet70-FS. Especially, the network incorporating with "CIF" has about 3% improvement for 1-shot setting.

**Comparison with other 3DFSL methods.** We further compare our proposed network with two recent 3D FSL works on ModelNet40_FS and ShapeNet70_FS. We follow the settings mentioned in [40] and [37], where Simpleshot is image-only low-shot recognition for 3D models' 2D RGB projections, and /w SB means shape-biased which uses auxiliary point cloud feature to learning a discriminative embedding space. For SSL3DFSL [37], We first train the based feature extractor with a self-supervised strategy proposed in [37], and then fine-tune the classifier with few samples. Table 7 summarizes the results on 5-way classification tasks, where our proposed baseline achieves much higher performance on both benchmarks.

### 5.6. Feature Visualization

We use t-SNE [19] to visualize the feature distribution at a 5way-5shot-15query setting on ModelNet40_FS with DGCNN [47] as backbone. Figure 3 (a) corresponds to the features of ProtoNet without the CIA module achieving an accuracy of 73.28%, (b) and (c) are the results of incorporating SCI and CIF, achieving 74.57% and 77.28% respectively, while (d) has better performance of 78.39% after equipped with CIA module. Note that the learning support and query features of ProtoNet [39] are dispersed with a huge distribution shift. After using the SCI module, the support features tend to move to the centrality to better differentiate from other classes, as shown in (b) and (d). Moreover, the distribution shift between support and query sets is mitigated, as shown in (c) and (d).

## 6. Conclusion

In this paper, we study 3D few-shot learning (FSL) in a systematic manner for the first time. We first empirically study and analyse the state-of-the-art 2D FSL algorithms in the context of point cloud classification and explore the influence of different backbone architectures. We then propose a strong baseline for benchmarking 3D FSL using ProtoNet with DGCNN. Furthermore, to alleviate the high intra-class variance and subtle inter-class differences among support set and query set, we also propose a plug-and-play Cross Instance Adaption (CIA) module consisting of a Self-Channel Interaction (SCI) module and a Cross-Instance Fusion (CIF) module, which can generate more discriminative features. For objective evaluation, we construct two benchmarks ModelNet40-FS and ShapeNet70-FS for 3D point cloud classification. Finally, extensive experiments show that the proposed CIA module has a significant performance improvement with different FSL algorithms and could be adapted well into different backbones.

**Acknowledgement:** This work is supported by National Natural Science Foundation of China (No. 62071127 and U1909207), Shanghai Pujiang Program (No.19PJ1402000), Shanghai Municipal Science and Technology Major Project (No.2021SHZDZX0103), Shanghai Engineering Research Center of AI Robotics and Engineering Research Center of AI Robotics, Ministry of Education in China, and Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project A18A2b0046)

## References


[1] Antreas Antoniou, Harrison Edwards, and Amos J. Storkey. How to train your MAML. In *ICLR*, 2019. 4

[2] Saifullahi Aminu Bello, Shangshu Yu, Cheng Wang, Jibril Muhmmad Adam, and Jonathan Li. deep learning on 3d point clouds. *Remote Sensing*, 12(11):1729, 2020. 2

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 2, 6

[4] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. *Proc. Int. Conf. Learn. Representations*, 2019. 2, 7

[5] Ali Cheraghian, Shafin Rahman, Dylan Campbell, and Lars Petersson. Transductive zero-shot learning for 3d point cloud classification. In *Proc. Winter Conf. on Appl. of Comp0 Vis.*, 2020. 1, 2

[6] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and QuocV Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, 2019. 2

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

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 3, 4, 7

[9] Yu Gao, Xintong Han, Xun Wang, Weilin Huang, and Matthew Scott. Channel interaction networks for fine-grained image categorization. In *AAAI*, pages 10818–10825, 2020. 5

[10] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3

[11] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3018–3027, 2017. 2

[12] Andrew E. Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449, 1999. 1, 3

[13] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *CVPR*, 2019. 2, 3, 4, 7

[14] Wenbin Li, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo. Revisiting local descriptor based image-to-class measure for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7260–7268, 2019. 3

[15] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in neural information processing systems*, pages 820–830, 2018. 3, 4, 8

[16] 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. 1

[17] Yongcheng Liu, Bin Fan, Gaofeng Meng, Jiwen Lu, Shiming Xiang, and Chunhong Pan. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *CVPR*, pages 5239–5248, 2019. 1, 3, 4, 8

[18] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *CVPR*, pages 8895–8904, 2019. 1, 3, 4, 8

[19] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. 8

[20] 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. 3

[21] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 3

[22] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731, 2018. 3, 6

[23] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1, 3, 4, 7, 8

[24] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016. 1

[25] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 3, 4, 8

[26] Yongming Rao, Jiwen Lu, and Jie Zhou. Spherical fractal convolutional neural networks for point cloud recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 452–460, 2019. 3

[27] Alexander J Ratner, Henry Ehrenberg, Zeshan Hussain, Jared Dunnmon, and Christopher Ré. Learning to compose domain-specific transformations for data augmentation. In *Advances in neural information processing systems*, pages 3236–3246, 2017. 2

[28] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2017. 3, 4, 7

[29] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *ICLR*, 2018. 2

[30] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. 3

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy,

Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 1

[32] Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. *arXiv preprint arXiv:1807.05960*, 2018. 3

[33] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ international conference on intelligent robots and systems*, pages 3384–3391. IEEE, 2008. 1

[34] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Mihai Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008. 1

[35] Justin Salamon and Juan Pablo Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017. 2

[36] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *ICLR*, 2018. 3, 4, 7, 8

[37] Charu Sharma and Manohar Kaul. Self-supervised few-shot learning on point clouds. *Advances in Neural Information Processing Systems*, 33, 2020. 3, 8

[38] Weijing Shi and Raj Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1711–1719, 2020. 1

[39] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 2, 3, 4, 6, 7, 8

[40] Stefan Stojanov, Anh Thai, and James M. Rehg. Using shape to categorize: Low-shot learning with an explicit shape bias. 2021. 3, 8

[41] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018. 3

[42] 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. 3

[43] 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 *CVPR*, 2018. 3, 4, 6, 7

[44] Federico Tombari, Samuele Salti, and Luigi Di Stefano. Unique shape context for 3d data description. In *Proceedings of the ACM workshop on 3D object retrieval*, pages 57–62, 2010. 1, 3

[45] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 2, 3, 6, 7

[46] Yan Wang, Wei-Lun Chao, Kilian Q Weinberger, and Laurens van der Maaten. Simpleshot: Revisiting nearest-neighbor classification for few-shot learning. *arXiv preprint arXiv:1911.04623*, 2019. 8

[47] 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. 3, 4, 6, 7, 8

[48] 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. 2, 6

[49] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015. 5

[50] Kai Yu, Wei Xu, and Yihong Gong. Deep learning with kernel regularization for visual recognition. In *Advances in Neural Information Processing Systems*, pages 1889–1896, 2009. 2

[51] Tan Yu, Jingjing Meng, and Junsong Yuan. Multi-view harmonized bilinear network for 3d object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 186–194, 2018. 1, 3

[52] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019. 1

[53] Yu Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 689–696. IEEE, 2009. 1

[54] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017. 1